Preparation of high-dimensional biomedical data with a focus on prediction and error estimation

Roman Hornung

München 2016

Preparation of high-dimensional biomedical data with a focus on prediction and error estimation

Roman Hornung

Dissertation
an der Fakultät für Mathematik, Informatik und Statistik
der Ludwig–Maximilians–Universität
München

vorgelegt von
Roman Hornung
aus Augsburg

München, den 19. September 2016

Erstgutachter: Prof. Dr. Anne-Laure Boulesteix

Zweitgutachter: Prof. Dr. David Causeur

Drittgutachter: Dr. Marie-Laure Martin-Magniette

Tag der mündlichen Prüfung: 15.12.2016

Summary

This thesis is based on three projects in which specific aspects of the preparation of high-dimensional biomedical data play a central role.

The first project is concerned with the question of whether it is necessary to include data preparation steps in cross-validation procedures in situations where such procedures are followed to estimate the prediction error of biomedical prediction rules. In practice many data preparation steps are, in most cases, not repeated within cross-validation on the training datasets, but rather performed beforehand on the entire dataset. Thereby, the training data and test data are not entirely separated, which, for some data preparation steps, can lead to a relevant underestimation of the prediction error. However, it is mostly unknown for which of these data preparation steps there is a danger of severe underestimation of this kind. In this project, first a general measure is developed to assess the magnitude of this underestimation. Next, this measure is applied to real datasets in extensive analyses in order to determine whether it is necessary to include two data preparation steps—normalization and principal component analysis—in cross-validation procedures.

In the second project an innovative method for batch effect adjustment is developed using which parts of a dataset that are systematically distorted, can be homogenized. This new method differs from others in that it removes both differences in terms of means and variances among the different dataset parts and corresponding differences in terms of the dependence structures of the variables. Using simulated data and real data, the performance of the new method is compared to that of popular alternative methods.

In the third project the possibility of employing batch effect adjustment and normalization to assimilate new data to the training data before prediction in order to achieve greater prediction accuracy is explored. In applications, the data to which a prediction rule is intended to be applied most often have, for various reasons, a slightly different distribution than the data on which the rule was derived. This often leads to increase in prediction error in practice. Therefore, it is desirable to adjust the distribution of new observations to that of the training observations prior to prediction. Here, both batch effect adjustment and normalization methods are suitable in partly modified forms; however, it is not clear which of these methods actually lead to greater prediction accuracy in practice. Therefore, with the help of a large number of real datasets, in the third project several of these methods are compared with respect to the prediction accuracies obtained from their use.

Zusammenfassung

Diese Arbeit basiert auf drei Projekten, in denen bestimmte Aspekte der Vorverarbeitung hoch-dimensionaler biomedizinischer Daten eine zentrale Rolle spielen.

Das erste Projekt befasst sich mit der Frage nach der Notwendigkeit des Einschlusses von Vorverarbeitungsschritten in Kreuzvalidierungsverfahren, wenn letztere zur Schätzung des Vorhersagefehlers von biomedizinischen Prädiktionsregeln verwendet werden. In der Praxis werden viele Vorverarbeitungsschritte zumeist nicht innerhalb der Kreuzvalidierung auf den Trainingsdatensätzen wiederholt, sondern vorher auf dem ganzen Datensatz durchgeführt. Für manche Vorverarbeitungsschritte kann dieses Vorgehen durch die damit verbundene Aufhebung der vollständigen Trennung von Trainings- und Testdaten zu einer relevanten Unterschätzung des Vorhersagefehlers führen. Es ist jedoch weitestgehend unbekannt, für welche Vorverarbeitungsschritte eine derartige Unterschätzung zu befürchten ist. Deshalb wird in dieser Arbeit zunächst ein allgemeines Maß zur Abschätzung des Ausmaßes dieser Unterschätzung entwickelt. Dieses Maß wird anschließend in ausführlichen Analysen auf echte Daten angewendet, um die Frage nach der Notwendigkeit des Einschlusses der beiden Vorverarbeitungsschritte Normalisierung und Hauptkomponentanalyse in Kreuzvalidierungsverfahren abschließend zu beantworten.

Im zweiten Projekt wird eine innovative Methode zur sogenannten Batcheffektentfernung entwickelt, anhand derer sich systematisch verzerrte Teile eines Datensatzes homogenisieren lassen. Die Besonderheit der neuen Methode gegenüber existierenden Alternativen besteht darin, dass sie sowohl Unterschiede in den Mittelwerten und Varianzen zwischen den verschiedenen Teilen des Datensatzes beseitigt als auch entsprechende Unterschiede in der Abhängigkeitsstruktur der Variablen. Die Performanz der neuen Methode wird anhand simulierter und echter Daten mit populären, alternativen Methoden verglichen.

Das dritte Projekt befasst sich schließlich mit der Möglichkeit, mit Hilfe von Batcheffektentfernung und Normalisierung vor der Anwendung von Prädiktionsregeln, die zu prädiktierenden Daten an die Trainingsdaten anzupassen, um eine größere Vorhersagegenauigkeit zu erzielen. In Anwendungen folgen die Daten auf die eine Prädiktionsregel angewendet wird, aus diversen Gründen zumeist einer leicht anderen Verteilung als die Daten auf denen sie gelernt wurde. Das führt dazu, dass der Vorhersagefehler von Prädiktionsregeln in der Praxis häufig erhöht ist. Deshalb ist es wünschenswert die Verteilung der Daten der zu prädiktierenden Beobachtungen an die Verteilung der Trainingsdaten anzupassen. Hierzu bieten sich Batcheffektentfernungsmethoden und Normalisierungsmethoden in teils abgewandelter Form an. Allerdings ist unklar, welche solcher Methoden in der Praxis tatsächlich zu einer größeren Vorhersagenauigkeit führen. Deshalb werden im letzten der drei Projekte einige dieser Methoden anhand einer großen Zahl echter Datensätze hinsichtlich der aus ihrer Verwendung jeweils resultierenden Vorhersagegenauigkeiten verglichen.

Acknowledgments

First and foremost, I would like to thank very warmly my thesis adviser Prof. Dr. Anne-Laure Boulesteix. She has pointed me in suitable directions and was always available for questions and helpful discussions. Moreover, she created an excellent encouraging collegial atmosphere in which I always felt secure during the preparation of the dissertation.

I would also like to extend my sincere thank to Prof. Dr. David Causeur. During my three research stays in Rennes I always felt very welcome. He pointed me in suitable directions as well, and in our many helpful discussions I was able to gain valuable insights.

Moreover, I am very grateful to my "periodical" colleagues at the Applied Mathematics Department at Agrocampus Ouest in Rennes, who amicably took care of me during my research stays in Britanny, for work as well as leisure.

Many thanks also to my colleagues at the Department of Medical Informatics, Biometry and Epidemiology for the great atmosphere.

Last but not least, I would like to thank my parents and my friends for their continued support and encouragement inside and outside of work.

Munich, in September 2016 *Roman Hornung*

Contents

1. Introduction

The analysis of modern, high-dimensional, biomedical data usually involves special preliminary data preparation steps, for example normalization or filtering by variance, which are performed before addressing the questions of thematic interest. With traditional medical datasets preprocessing usually is more straightforward and easier to conduct. Preliminary steps in this field may involve standardizing the variables or imputing missing variables. Given the increasingly important role of high-dimensional biomedical data in medical research, data preparation can be expected to gain attention in the scientific literature.

This thesis consists of three parts, in each of which, preparation of high-dimensional biomedical data plays a central role. In the first part the impact of excluding specific data preparation from cross-validation on the estimated error is analyzed with the aid of the cross-validation incompleteness impact measure (CVIIM) developed specially for this purpose. In the second part, a new batch-effect adjustment method for high-dimensional data is presented, which, in addition to adjusting for location-and-scale batch effects, adjusts for batch effects evident in the correlation structures within the batches. In the third part, a compendium of microarray datasets is used to determine the extent to which, and under which circumstances, cross-study prediction can be improved with the aid of addon batch effect adjustment and addon normalization.

Background

In the following, several terms which play a crucial role in the thesis are explained.

High-dimensional data

This term describes datasets featuring more variables than observations. *High-throughput datasets* are specific types of high-dimensional datasets featuring many—often many thousands of—biomolecular variables. In this context, each variable often corresponds to the behavior of a certain gene. Traditional statistical analysis methods such as linear or logistic regression are not applicable to high-dimensional data.

However, in the last few decades many methods have been developed that are specialized for this data format. Many of these methods were used in the analyses presented in this thesis.

Data preparation step

In general, this term is used for any analysis step performed before conducting analysis steps that deliver the final result of the study. However, in this thesis there is a difference between data preparation steps performed observation by observation, such as background correction in microarray data, and data preparation steps which use information across observations. Examples of the latter steps are quantile normalization of microarray data, batch effect adjustment (see below), and variable selection. In this thesis only steps of the second kind are considered, which is why in the following, the term *data preparation step* refers to this kind of data preparation.

Normalization

Various factors influencing raw high-throughput data lead to differences among observations. Such differences are due to elements beyond the biological signal of interest. Therefore, the distributions of the raw values generally are made similar across observations by employing a normalization method. Due to differences among the various high-throughput data structures, each data type requires the use of a specific normalization method. An important method is the quantile normalization method (Bolstad et al.; 2003), which is used in this thesis. After employing this method, the empirical distributions of the data values are identical across the different observations.

Prediction rule

A central application field of high-throughput data is the prediction of disease states of patients. To this end, prediction rules are used. A prediction rule is an algorithm that uses the values of a number of variables in a patient (*covariates*) to predict the unknown value of a specific phenotype variable of interest (*target variable*). Such prediction rules are obtained, or *learned*, using the data on a series of patients for whom the covariate values and the values of the target variable are known. These data are commonly denoted as *training data*. The algorithm that is used for learning the prediction rule is specific to the latter. This thesis is concerned with prediction rules for high-dimensional data with binary target variables. A prediction rule of this kind may, for example, be obtained by applying the random forest (RF) algorithm (Breiman; 2001), which involves constructing a large number of classification trees

using the training data. The application of the learned prediction rule then involves applying the classification trees constituting the constructed RF to the data on a new patient and aggregating the results to obtain a prediction of his/her value of the target variable.

When applying prediction rules in practice one is confronted with the raw covariate data on new patients. Therefore, here, as a preliminary step, any data preparation steps that have been taken while learning the prediction rule have to be taken for the data on new patients as well. It is impossible to obtain a prediction based on the patients' raw covariate data, because the learning algorithm which delivered the final prediction rule has used the form of the training data obtained after preparation. Due to the widespread use of data preparation steps, this issue can be expected to be equally widespread in practice. However, this issue seems widely overlooked in the scientific literature. In this thesis the following definition of prediction rules is given: A prediction rule comprises not only the algorithm specific to the method outputting the predictions based on the preprocessed data, for example the RF, but also any data preparation steps to be performed before the preprocessed data can be committed to this algorithm. In accordance with this definition, the algorithm used for learning such a prediction rule comprises all data preparation steps.

Estimation of the error of a prediction rule

Before applying a prediction rule, it is important to assess how accurately it can predict the values of the target variable of independent patients. If the error frequency of a prediction rule is too high, it should not be applied in practice. This is because doctors and patients would be misinformed too frequently if such a rule were used for diagnosis. A naïve and very problematic approach to estimate the error frequency would be the following: Apply the prediction rule to the training data with which it was learned and calculate the frequency of which the prediction rule delivers an incorrect prediction for this data. It is well-known that this can lead to a severe underestimation of the error frequency to be expected for independent data. The reason for this is that the prediction rule tends to be better adjusted to its training data than to the independent data for which it is intended. Therefore, as a general rule the data used to estimate the error frequency, the *test data*, should be different from those used to learn the prediction rule. Here it is possible to discern two cases: 1) The test data are distinct from the training data but originate from the same dataset as the training data, referred to as *internal validation*; 2) The test data stem from an independent dataset, referred to as *external validation*. As the data to which the prediction rule is intended to be applied, usually originate from an entirely different

dataset than that of the training data, external validation results in more realistic error estimates than internal validation. As data from different datasets other than from where the training data originate, generally behave more differently than those from the same datasets, external validation results in higher error estimates (see e.g. Bernau et al. (2014)).

K-fold cross-validation

The most prominent method for estimating the error frequency to be expected in internal validation is K-fold cross-validation or short *cross-validation* (CV). High-throughput datasets are often quite small; therefore, splitting the dataset into training data and test data usually leads to at least one of the following two problems: 1) The test dataset is too small to provide an acceptable accuracy of the error estimate; 2) The training dataset is too small to ascertain a level of prediction performance similar to that which results when using the entire dataset as a training dataset. K-fold CV addresses these issues first by re-performing the error estimation for different splits of the entire dataset into training data and test data and second by choosing a large enough amount of training data. Precisely, K-fold CV is conducted as follows: 1) Split the dataset randomly into K (approximately) equally sized parts, denoted as *folds*; 2) For $k \in \{1, \ldots, K\}$: estimate the error frequency using fold k as test data and the rest of the dataset as training data; 3) Take the average of the error estimates from 2). To reduce further the variance of the error estimation, in practice K-fold CV should be repeated multiple times. Although K-fold CV is less variable than error estimates obtained when using single splits into training data and test data, its variance is still high (Efron and Tibshirani; 1997). Ostensibly, no widely accepted guidelines exist for choosing the number of folds K.

Batch effects

Traditional clinical data most often directly mirror the biological phenomena of interest such as age or sex. By contrast, the data values from microarray-based data types constitute measurements of the respective biological phenomena such as gene expression. These measurements are unwantedly influenced by external conditions. There are numerous external factors that can influence microarray data, such as the laboratory in which the data are produced, the technician treating the data and even the time of data generation. As a result, data of observations which have the same or similar biological characteristics and, thus, should behave similarly, can be distributed very differently when stemming from different datasets. Such effects gen-

erally are known as *batch effects*. The totality of data from a specific source is denoted as a *batch*. Batch effects lead to limited comparability of data from different sources, which can negatively influence analyses being performed using the combined data. In contrast to microarray-based data types, which require measuring the underlying biological phenomena, more modern high-throughput data types such as RNA-Seq data allow direct investigation of the underlying biological phenomena. Nevertheless, data collection in the case of these modern high-throughput data types is prone to error and batch effects have been found to be a problem with these data as well (Hansen and Irizarry; 2012).

In this thesis, the definition of batch effects comprises not only effects caused by external conditions of the data generation process but also by differences in data cohorts due to differing study populations, as long as these differences do not affect the biological phenomena to be studied with the aid of the data.

Batch effect adjustment

Given that batch effects can influence analyses negatively, it is desirable to remove such effects beforehand by assimilating the distributions of the data across the batches. A method with such an aim is called a *batch effect adjustment* or *batch effect removal* method. A number of such methods exist. Some of them are very simple, for example, zero-mean centering of the variable values in the batches, while others are more sophisticated, for example, adjusting for location-and-scale differences across batches as performed by ComBat (Johnson et al.; 2007). Note that batch effect adjustment is a data preparation step.

Cross-study prediction

As mentioned above, a prediction rule is applied in practice to data stemming from entirely different sources than those of the training data. This procedure is referred to as *cross-study prediction* in this thesis.

Contents

In this section the three projects presented in the thesis are described in order to provide an initial overview of the work.

Chapter 2: Measuring the impact of CV incompleteness with respect to data preparation steps

As stated above, in CV the prediction rule is learned repeatedly using different subsets of the available dataset. The learning process of obtaining a prediction rule also comprises the data preparation steps involved according to the definition of prediction rules given above. Therefore, from a formal standpoint, data preparation should be re-performed using the training data in each iteration of the CV. Performing data preparation on the entire dataset before performing error estimation is equivalent to learning part of the prediction rule using both training data and test data. This can have negative implications. Above it was stated that the prediction rule is better adjusted to the data on which it was learned. If these data comprise the test data for part of the learning process, as done when performing data preparation on the entire dataset, it can be expected that the test data are better adjusted to the prediction rule than truly independent data are. Again, this may lead to underestimation of the error frequency. Chapter 2 is concerned with the latter phenomenon.

In fact, for various reasons, it is common practice to perform certain data preparation steps, for example, quantile normalization, using the entire dataset before CV. The extent of underestimation of the error frequency resulting from this, that is, the impact of such an *incomplete CV*, differs among data preparation steps. Steps making use of the target variable more frequently are associated with a stronger impact on the estimated error through incomplete CV. However, as will be shown in the analyses in Chapter 2, there also are steps which use only the covariate values but nevertheless should not be performed on the entire dataset before error estimation to warrant against underestimating the error frequency.

The work presented in Chapter 2 resulted from the following three considerations: 1) The impact of incomplete CV depends on the data preparation step under consideration; 2) It is not clear which data preparation steps have to be performed within CV to prevent a relevant optimistic bias; 3) Someone confronted with a dataset cannot be expected to decide for his/her specific analysis whether or not he/she should include a data preparation step in CV. These considerations necessitate the formulation of empirically based guidelines for individual data preparation steps that state whether or not these steps generally can be conducted on the entire dataset before CV. When

formulating such guidelines based on real datasets, it is convenient, if not required, to use a measure of the impact of incomplete CV. In Chapter 2 such a measure is provided with the *CV incompleteness impact measure* (*CVIIM*) which depends on the distribution underlying the dataset used for estimating it. Therefore, in addition to CVIIM, with the *global CVIIM* a measure is provided that does not depend on a specific data distribution. The estimation of this measure requires several datasets.

With the aid of CVIIM and global CVIIM and through using large collections of real high-throughput datasets, guidelines for the following two commonly used data preparation steps are developed: normalization and principal component analysis (PCA). These guidelines state that while normalization can be performed beforehand on the entire dataset, PCA should be included in CV. Corresponding empirically based guidelines ostensibly were previously available only for the data preparation step "supervised variable selection". In addition to the guidelines for normalization and PCA, preliminary results obtained for other common data preparation steps are provided.

Chapter 3: Combining location-and-scale batch effect adjustment with data cleaning by latent factor adjustment

Due to the wide variety of factors leading to batch effects, the latter are equally diverse in nature. Therefore a batch effect adjustment method intended to work well for various datasets should do at least one of the following: 1) address coarse types of batch effects that are present in all datasets affected by batch effects. Examples of methods of this kind are mean-centering, standardization, ratio-A, ratio-G (Luo et al.; 2010), and ComBat; 2) adjust to the specific kind of batch effects of the dataset being analyzed. An example of a method of this kind is surrogate variable analysis (SVA) (Leek and Storey; 2007).

Methods of the first kind may miss important characteristics of batch effects present in the specific dataset being analyzed, for example, sophisticated dependence structures induced by batch effects. By contrast, methods of the second kind may miss simple batch effect structures which can easily be addressed, for example, mean-differences of the variables across batches. In Chapter 3 a method is presented with *FAbatch* that addresses coarse features of batch effects and adjusts to sophisticated batch effect patterns specific to the individual dataset being studied. FAbatch represents an extension of ComBat, that is, adjustment for location-and-scale differences of the variables across batches. With FAbatch, in addition to location-and-scale adjustment, the data are adjusted for latent factor influences associated with batch effects within the individual batches. This has the effect of reducing excess heterogeneity

within batches, that is, variations in the observations not attributable to the biological signal of interest are addressed, which is very similar to the approach followed by SVA. However, in contrast to FAbatch, SVA was developed for situations where it is not known which observation belongs to which batch. The primary goal of the adjustment for latent factor influences performed by SVA is to remove heterogeneity resulting from observations belonging to different batches. The adjustment for latent factor influences performed by FAbatch within batches, by contrast, should capture sophisticated properties of the batches specific to the dataset being studied. In the estimation of the latent factor models, it is important that the biological signal of interest be protected. Otherwise, not only unwanted differences among the observations attributable to batch effects would be removed, but also would desired differences attributable to the biological signal. SVA uses a specific procedure for protecting the biological signal. This procedure can lead to a dangerous exaggeration of the biological signal, as will be discussed in detail in Chapter 3. By contrast, with FAbatch the signal exaggeration is mitigated by using predicted class probabilities instead of the actual classes in the protection of the biological signal.

In Chapter 3, using a large collection of real high-throughput datasets, FAbatch is extensively compared to commonly used competitors with respect to several metrics. These metrics measure either the homogeneity of the data across batches after batch effect adjustment or the performance of analyses performed using the batch effect adjusted data. Here, FAbatch proves to be equal to the other methods in many situations and superior in some situations.

Moreover, for illustrative purposes FAbatch and its competitors are applied in a prediction context. In prediction, batch effect adjustment can be employed to render data to which the prediction rule is to be applied more similar to the training data. In this thesis, reference is made to *addon batch effect adjustment* when applying batch effect adjustment in this way. Addon batch effect adjustment is properly defined in Chapter 3 and will be at the center of Chapter 4.

Chapter 4: Improving cross-study prediction through addon batch effect adjustment or addon normalization

When obtaining prediction rules, it is implicitly assumed that the data to which the rule is intended to be applied follow the same distribution as that underlying the training data. However, this assumption is—due to batch effects—not given in many cases when applying prediction rules in practice. Consequently, the true error of a prediction rule will be greater than the error estimate obtained through CV. Given the need for similarity between the data to which a prediction rule is applied and the

training data, it is a natural goal to transform the new data before prediction in such a way that they follow the same distribution as the training data. In the following the term *test data* will be used for the data to which a prediction rule is applied. Note that this term also is used for the data used to estimate the error rate of a prediction rule. However, there is no likelihood of confusion between the two meanings because the correct meaning of this term will be clear from the respective context.

As noted above, batch effect adjustment can be performed to transform the test data in order to adjust them to the training data. Many batch effect adjustment methods can be applied to this end without the need for alteration. Others have to be adjusted in such a way that the training data are not changed in the process of considering varying test datasets. However, the latter is not a difficult task using the general blueprint for determining the addon procedure for a specific batch effect adjustment method introduced in Chapter 3.

The aim of batch effect adjustment is to assimilate the distributions of the individual variables across different data subparts. The aim of normalization is quite similar: to adjust the marginal distributions of the data values across observations. Thus, normalization also can be used in adjusting the test data to the training data. Here, as with batch effect adjustment, it has to be warranted that the training data are not altered when transforming the test data. A normalization procedure that fulfills this is termed *addon normalization* in the following. The only difference between a normalization procedure and an addon normalization procedure is the following: In an addon normalization procedure, those parameters involved which are not observation-specific are estimated using the training data only, sparing out the test data. The addon normalization procedure for robust multi-array average (RMA) normalization was developed by Kostka and Spang (2008). Addon procedures can be determined for any data preparation steps that involve considering information across observations (see Chapter 2).

From an intuitive point of view, it is clear that both addon batch effect adjustment and addon normalization should lead to better prediction results when the test data follow a different distribution from that of the training data before adjustment. However, in Chapter 4 a large study of real data is conducted to investigate the value of these procedures in a prediction context for the following two main reasons: 1) Even though an improvement is expected, it is not clear whether there is indeed a notable gain in prediction performance by applying these procedures in practice; 2) There are various batch effect adjustment methods and it is not clear whether all of them lead to an improvement. In this extensive study 25 real microarray datasets are considered, where "sex" is used as a binary target variable. For each setting studied, all 25 datasets are considered once as a training dataset and all other 24 datasets as

test sets.

From the results of this extensive study of real data the following conclusions can be drawn: 1) There is an improvement both by addon batch effect adjustment and addon normalization; 2) Addon batch effect adjustment generally is more effective than addon normalization; 3) There is no advantage of combining addon batch effect adjustment and addon normalization over using addon batch effect adjustment alone; 4) While addon batch effect adjustment is recommendable in principle, it should be applied only when the test datasets are not too small and when the distribution of the target variable is similar in training data and test data; 5) Only those batch effect adjustment methods that address coarse batch effects seem to be appropriate. Other kinds of methods impaired performance in the analyses.

Publications and submitted articles

The works described in Chapters 2, 3, and 4 have been published in the journals BMC Medical Research Methodology, BMC Bioinformatics, and Bioinformatics, respectively:

- Hornung, R., Bernau, C., Truntzer, C., Wilson, R., Stadler, T., and Boulesteix, A.-L. (2015). A measure of the impact of CV incompleteness on prediction error estimation with application to PCA and normalization. BMC Medical Research Methodology, 15, 95.

- Hornung, R., Boulesteix, A.-L., and Causeur, D. (2016). Combining location-and-scale batch effect adjustment with data cleaning by latent factor adjustment. BMC Bioinformatics, 17, 27.

- Hornung, R., Causeur, D., Bernau, C., and Boulesteix, A.-L. (2016). Improving cross-study prediction through addon batch effect adjustment or addon normalization. Bioinformatics, doi: 10.1093/bioinformatics/btw650.

Moreover, the R package *bapred* was developed in the context of the project presented in Chapter 3. This package allows batch effect adjustment and addon batch effect adjustment using the new method FAbatch and various other common batch effect adjustment methods. Further, it provides various metrics for assessing the success of batch effect adjustment. The package was extended in the context of the project presented in Chapter 4 by addon quantile normalization and addon RMA normalization.

Contributions to chapters

Chapters 1 and 5 were written by me without any contributions by others. Moreover, I was the main contributor of the manuscripts of the papers underlying Chapters 2 to 4.

I wrote, by far, the greatest part of the manuscript of the paper underlying Chapter 2. The co-authors helped with revising the manuscript, setting up the study design of the empirical study presented in section 2.2.8 and implementing the simulation study presented in Appendix A.1 and in section 2.4.3.

The manuscript of the paper underlying Chapter 3 was written also almost completely by me. The co-authors helped with revising the manuscript, setting up the design of the empirical comparison study presented in section 3.2.4 and contributed to the concept of the FAbatch method.

I also wrote the manuscript of the paper underlying Chapter 4 almost completely by myself. The co-authors helped with revising the manuscript.

With the exception of the help I received in implementing the simulation study for the manuscript of the paper underlying Chapter 2, I performed all analyses presented in the thesis by myself.

2. Measuring the impact of CV incompleteness with respect to data preparation steps

2.1. Background

In supervised statistical learning, it is widely recognized that prediction models should not be constructed and evaluated using the same dataset. While the training dataset is used for all steps towards obtaining the prediction rule, the test dataset is used to evaluate its prediction error and, ideally, should not be at all involved in the training phase. CV and related procedures involve considering several divisions into training data and test data and averaging the estimated prediction errors of the respective prediction rules constructed in each iteration (see also Chapter 1). In this chapter K-fold CV is used, but all ideas and procedures can be extended to other resampling techniques used for prediction error estimation.

In the following the term *incomplete CV* (Simon et al.; 2003) refers to CV procedures in which some analysis steps are performed beforehand using the entire dataset. With incomplete CV, at each iteration the excluded fold acting as test data may affect the prediction rule derived, since it was used preliminarily for data preparation which contradicts the principle of test data requiring perfect separation (Daumer et al.; 2008). In contrast, if all steps leading to the prediction rules are performed in each CV iteration using only the corresponding training set, the CV procedure is *full CV*.

The problems resulting from incomplete CV have been studied extensively with regard to preliminary variable selection for classification based on high-dimensional microarray data (Simon et al.; 2003; Ambroise and McLachlan; 2002; Wood et al.; 2007; Zhu et al.; 2008). If performed *before* splitting the dataset into K folds, supervised variable selection often leads to strongly, downwardly biased error estimates. The now widely adopted procedure to avoid this problem involves conducting anew the variable selection step in each CV iteration using the training dataset only (Si-

mon et al.; 2003; Ambroise and McLachlan; 2002), that is, considering it part of the classifier construction process. Similarly, it has been suggested that parameter tuning should be performed using the training dataset only (Varma and Simon; 2006; Bernau et al.; 2013; Boulesteix and Strobl; 2009). However, the bias resulting from incomplete CV with respect to parameter tuning ostensibly has never been investigated.

Variable selection and parameter tuning are—by far—not the only procedures often performed in practice before CV. For example, raw data from high-throughput biological experiments such as microarrays have to be normalized before high-level analyses such as predictive modeling can be conducted. The selection of features exhibiting considerable variability across the observations is another example of a data preparation step often performed when analyzing microarray data. Further examples relevant to any type of data include imputation of missing values, dichotomization, and non-linear transformations of the features. Preparation steps are not limited to these few examples. The analysis of increasingly complex biomedical data (including, e.g., imaging or sequencing data) requires the use of evermore sophisticated preprocessing steps for making raw data analyzable. Note again that the question of the impact of CV incompleteness is not relevant to those steps which prepare the observations independently of each other, such as background correction for microarray data.

Although it is known that incomplete CV is problematic in the case of variable selection, it is unknown which other preparation steps lead to underestimation of the prediction error if performed before splitting the dataset into K folds. To date there seems to be no consensus on whether it is necessary to include all steps in CV: Some authors postulate that all steps are required to be included (Westerhuis et al.; 2008), which seems to be done rarely, regardless; others only suggest this procedure for variable selection (Ambroise and McLachlan; 2002) or, generally, for supervised steps (Hastie et al.; 2009).

Some practical problems which deter researchers from performing full CV include the computational effort often needed to repeat time-intensive preparation steps, the fact that some preparation steps such as variable selection are sometimes conducted "in the lab" before the data are given to the statistician (Zhu et al.; 2006), and the lack of (user-friendly) implementations of addon procedures allowing the adequate preparation of the excluded fold when the preparation step has been conducted using the training folds only (see section 2.2.2 for more details on addon procedures). Another practical problem occurs in the context of genotype calling in genetic association studies: It is common practice to use not only the entire dataset of interest, but also additional datasets to improve genotype calling accuracy.

In the context of high-dimensional data, two further important preparation steps

often performed using the entire dataset are dimension reduction procedures such as PCA and normalization—for example normalization using the RMA method (Irizarry et al.; 2003) for microarray gene expression data. It is not clear whether the prediction error estimate that results is optimistically biased if one applies these two methods to the entire dataset before splitting the data into K folds. In an effort to answer this question, a measure is presented in this chapter which enables the quantification of the impact of incomplete CV with regard to steps of interest, the *CV incompleteness impact measure* (CVIIM). It is based on the ratio of the CV prediction error which results when the preparation steps investigated are applied only once using the entire dataset to the CV prediction error which results when they are incorporated into CV. The latter means that, within CV, in addition to performing the investigated preparation steps on each training dataset anew they are applied to the excluded fold through addon procedures.

The goal of this chapter is two-fold: (i) to present the new measure CVIIM, which is intended to be used by methodology researchers or statisticians working on statistical learning applications to determine whether a particular preparation step should, in general, be trained in each CV iteration successively or whether it can be performed safely as a preliminary step on the entire dataset without generating a relevant optimistic bias; and (ii) to apply this new measure to answer this question for two important preparation steps, PCA and normalization, in order to provide corresponding guidelines for these steps.

This chapter is structured as follows: In section 2.2 first the microarray gene expression datasets used in the empirical studies described below, the concept of addon procedures, and the two methods, normalization and PCA, are presented. Then CVIIM is introduced and its use and behavior in the well investigated case of variable selection are briefly illustrated using four datasets. Finally, the designs of the studies on the impact of CV incompleteness with respect to normalization and PCA are described. In section 2.3 the results of these studies are presented. In section 2.4 preliminary results obtained for other data preparation steps are presented and further issues are discussed. In section 2.5 the main conclusions drawn from this chapter are summarized.

2.2. Methods

2.2.1. Data material

A wide range of publicly available, high-dimensional, mostly transcriptomic datasets were used in the real data analyses. See Table 2.1 for an overview.

Study	Label/ acc. number	Num. of observ.	Num. of variables	Prop. smaller class	Data type	ID
Normalization	E-GEOD-10320	100	22283	0.42	transcription	1
Normalization	E-GEOD-47552	74	32321	0.45	transcription	2
Normalization	E-GEOD-25639	57	54675	0.46	transcription	3
Normalization	E-GEOD-29044	54	54675	0.41	transcription	4
Normalization	E-MTAB-57	47	22283	0.47	transcription	5
Normalization	E-GEOD-19722	46	54675	0.39	transcription	6
Normalization	E-MEXP-3756	40	54675	0.50	transcription	7
Normalization	E-GEOD-34465	26	32321	0.35	transcription	8
Normalization	E-GEOD-30174	20	54675	0.50	transcription	9
Normalization	E-GEOD-39683	20	32321	0.40	transcription	10
Normalization	E-GEOD-40744	20	20706	0.50	transcription	11
Normalization	E-GEOD-46053	20	54675	0.40	transcription	12
PCA	E-GEOD-37582	121	48766	0.39	transcription	13
PCA	ProstatecTranscr	102	12625	0.49	transcription	14
PCA	GSE20189	100	22277	0.49	transcription	15
PCA	E-GEOD-57285	77	27578	0.45	DNA methyl.	16
PCA	E-GEOD-48153	71	23232	0.48	proteomic	17
PCA	E-GEOD-42826	68	47323	0.24	transcription	18
PCA	E-GEOD-31629	62	13737	0.35	transcription	19
PCA	E-GEOD-33615	60	45015	0.35	transcription	20
PCA	E-GEOD-39046	57	392	0.47	transcription	21
PCA	E-GEOD-32393	56	27578	0.41	DNA methyl.	22
PCA	E-GEOD-42830	55	47323	0.31	transcription	23
PCA	E-GEOD-39345	52	22184	0.38	transcription	24
PCA	GSE33205	50	22011	0.50	transcription	25
PCA	E-GEOD-36769	50	54675	0.28	transcription	26
PCA	E-GEOD-43329	48	887	0.40	transcription	27
PCA	E-GEOD-42042	47	27578	0.49	DNA methyl.	28
PCA	E-GEOD-25609	41	1145	0.49	transcription	29
PCA	GSE37356	36	47231	0.44	transcription	30
PCA	E-GEOD-49641	36	33297	0.50	transcription	31
PCA	E-GEOD-37965	30	485563	0.50	DNA methyl.	32

Table 2.1.: Overview of the datasets used in the studies on normalization and PCA. The following information is given: accession number, number of observations, number of variables, proportion of observations in the smaller class, data type. NCBI GEO accession numbers have the prefix `GSE`.

With the exception of `ProstatecTranscr` all datasets were downloaded from the ArrayExpress database (`www.ebi.ac.uk/arrayexpress`) (Kolesnikov et al.; 2015) or the NCBI GEO database (`www.ncbi.nlm.nih.gov/geo`) (Barrett et al.; 2013). All datasets feature a binary target variable and are of human origin. Details on the biological background of the datasets may be obtained online with the respective accession numbers available in Table 2.1 and via the R scripts written for the preparation of the individual datasets for analysis. The latter are available at `http://www.ibe.med.uni-muenchen.de/organisation/mitarbeiter/070_drittmittel/hornung/cviim_suppfiles/index.html` and can be used to download and prepare the individual datasets automatically. The dataset `ProstatecTranscr` appeared in Singh et al. (2002) and is available in the form of an `Rda`-file at the above link as well. Here, R scripts also are provided for reproducing all analyses presented in this chapter.

In the search for suitable datasets those which featured a strong class imbalance or would have been difficult to handle from a computational point of view were excluded.

2.2.2. Addon procedures

In this section a brief overview is given of the concept of addon procedures. When a data preparation step has been conducted on the training data only, the test data must be prepared equivalently: To not do so might render the test data nonsensical with regard to, or even incompatible with, the prediction rule derived from the training data. A naïve but straightforward procedure in the case of steps which do not involve the response variable (*unsupervised* steps) such as normalization (see section 2.2.3) is to prepare the test data completely independently, that is, without using any information from the preparation of the training data. For the prediction of external data such a separate data preparation procedure may be suitable for some steps when the external data behave very differently from the training data: By separate processing the data preparation procedure can adjust itself to the peculiarities of the external data (see e.g. Bin et al. (2014)). However, in many situations this approach may lead to greater prediction error particularly in the case of small test datasets due to the larger variance of the output of preparation steps. Test datasets of size one (corresponding to, for example, patients examined one at a time) are an extreme case in which this approach is even unfeasible. Moreover, for some preparation steps such as variable filtering by variance this naïve approach cannot be applied because it would lead to the selection of different variables in the training datasets and test datasets and thus make the application of the prediction rule impossible.

Another straightforward approach is to "train" the preparation step on the training

data and to use the output of the preparation step to prepare the test data. In the following, this is referred to as an *addon procedure*. This term originally was introduced in the specific case of normalization for microarray data (Kostka and Spang; 2008) but is employed here for all types of data preparation steps. In this thesis, the following definition is used: An addon procedure for a data preparation step prepares an observation in the test data precisely as it would prepare a corresponding observation in the training data, where, however, the parameter estimates involved have been obtained exclusively on the training data. This, of course, excludes parameters specific to the individual observations. These parameters are still estimated using the data of the corresponding observations. Note that by *performing* a preliminary step the following is meant in the context of this chapter: 1) Conduct the preparation step on the data considered; 2) Store all information necessary for addon preparation of new observations. Addon procedures are trivial in some cases, for instance, that for dichotomization according to cutpoints determined from the training data (one simply uses the training-data-derived cutpoint to dichotomize the test data) or that of variable selection (selecting precisely those variables in the test data which were selected based on the training data). However, in other cases, such as normalization of microarray data or imputation of missing values, this task is more complex.

2.2.3. (Addon) normalization

Normalization of microarray data essentially is the transformation of the data in such a way as to eliminate or reduce systematic differences among observations unrelated to biological differences. In this chapter two methods of microarray data normalization are considered: 1) RMA and 2) RMA where the quantile-normalization step is expanded by variance stabilization normalization (VSN) (Huber et al.; 2002) without calibration (RMAglobalVSN) (Huber; 2014). RMA consists of three steps: 1) background correction, 2) quantile normalization (Bolstad et al.; 2003), and 3) summarization. Background correction and summarization are performed on an array-by-array basis, which is why no addon strategies are necessary for these procedures. The quantile normalization step is performed conceptionally as follows: Let x_{sort,i^*j} be the j-th smallest variable value of array i^*. Then for each array $i \in \{1, \ldots, n\}$ the j-th smallest value is determined and the average $\bar{x}_{\text{sort},j}$ over these n values taken. Finally x_{sort,i^*j} is replaced by $\bar{x}_{\text{sort},j}$. By performing this procedure for all variable values, the empirical distributions of all arrays become equal. When normalizing the test observations using addon quantile normalization (Kostka and Spang; 2008) the averages of the j-th smallest values are obtained using the training data only, that is, excluding the corresponding test observations. As a consequence the scale of the normalized

test observations is consistent with that of the normalized training observations without the latter having been changed during the procedure. VSN transforms the gene expression values in such a way that the variance of the differences between values of different observations is rather constant across the entire intensity range. In the vignette of the Bioconductor package `vsn`, Huber (2014) presents a version of VSN in which no calibration is performed, that is, only a global variance stabilization transformation is conducted. In contrast to standard VSN this procedure does not involve any observation-specific parameters, so it is possible to determine an addon procedure: The global VSN parameters estimated on the training data are used to transform the test data.

2.2.4. (Addon) principal component analysis

PCA is an unsupervised dimension reduction method commonly used in the context of high-dimensional data analysis. The principal components are calculated using a singular value decomposition (SVD) of the centered data matrix. The addon procedure is as follows: 1) Center the values of each variable by subtracting the corresponding variable mean estimated from the training data; 2) Multiply the matrix resulting from 1) by the PCA loading matrix derived from the training data to obtain the principal components. The principal components with highest variance can be viewed as a summary of the data in fewer dimensions and often are used in practice for graphical representation of the data. In the context of classification using high-dimensional data, it is common to fit a prediction rule with a prediction method such as discriminant analysis using principal components as predictors instead of the original variables (Dai et al.; 2006).

2.2.5. The CV incompleteness impact measure (CVIIM)

In the following, CVIIM, the new measure to determine the extent of bias induced by incomplete CV with respect to a data preparation step of interest, is presented. Let $\boldsymbol{s}$ be the available dataset from which a prediction rule is to be derived. $\boldsymbol{s}$ is assumed to be an *i.i.d.* sample of size n with observations drawn from the distribution P, where P is the joint distribution of predictors and a response variable. Note that the assumption of *i.i.d.* observations made here is owing to the fact that this chapter is concerned with CV, that is, dataset internal validation. With external validation this assumption generally is not appropriate. Further, let $e_{full,K}(\boldsymbol{s})$ be the prediction error estimated by full K-fold CV, that is, all steps leading to the prediction rule, including data preparation steps, are performed anew at each CV iteration

based on the training dataset only. Similarly, let $e_{incompl,K}(\boldsymbol{s})$ be the prediction error estimated by incomplete K-fold CV, that is, the data preparation step(s) of interest is/are performed before CV, using the entire dataset. For simplicity of notation, it is assumed that $e_{full,K}(\boldsymbol{s})$ and $e_{incompl,K}(\boldsymbol{s})$ are obtained by averaging over a large number of CV runs, that is, over a large number of random partitions, and can thus be treated as deterministic.

For $\boldsymbol{S} \sim P^n$, the measure CVIIM is defined as:

$$\text{CVIIM}_{P,n,K} := \begin{cases} 1 - \dfrac{\mathbb{E}[e_{incompl,K}(\boldsymbol{S})]}{\mathbb{E}[e_{full,K}(\boldsymbol{S})]} & \text{if } \mathbb{E}[e_{incompl,K}(\boldsymbol{S})] < \mathbb{E}[e_{full,K}(\boldsymbol{S})] \\ & \text{and } \mathbb{E}[e_{full,K}(\boldsymbol{S})] > 0 \\ 0 & \text{otherwise.} \end{cases} \tag{2.1}$$

Note that $\text{CVIIM}_{P,n,K}$ is defined as a theoretical quantity, not calculable, but possible to estimate from real data. It is estimated simply by replacing the expected CV errors with their empirical counterparts $e_{incompl,K}(\boldsymbol{s})$ and $e_{full,K}(\boldsymbol{s})$:

$$\text{CVIIM}_{\boldsymbol{s},n,K} := \begin{cases} 1 - \dfrac{e_{incompl,K}(\boldsymbol{s})}{e_{full,K}(\boldsymbol{s})} & \text{if } e_{incompl,K}(\boldsymbol{s}) < e_{full,K}(\boldsymbol{s}) \\ & \text{and } e_{full,K}(\boldsymbol{s}) > 0 \\ 0 & \text{otherwise.} \end{cases} \tag{2.2}$$

Clearly, $\text{CVIIM}_{P,n,K} \in [0, 1]$. The same holds for the estimator $\text{CVIIM}_{\boldsymbol{s},n,K}$. CVIIM is based on the ratio of the incomplete CV error to the full CV error, which is more revealing than their difference as a measure of the impact of CV incompleteness. Indeed, the latter would depend heavily on the value of the error (large error values leading to large differences), as suggested by the results shown in section 2.4.1 and by the simulation presented in section 2.4.3 and in Appendix A.1. Truncation at zero prevents CVIIM from being negative in the unlikely case that the incomplete CV error is larger than the full CV error. A large value of CVIIM indicates that CV incompleteness results in a large underestimation of the prediction error.

The discrepancy between $e_{incompl,K}(\boldsymbol{s})$ and $e_{full,K}(\boldsymbol{s})$ depends on the extent to which the specific preliminary step conducted on the entire dataset increases the homogeneity of the covariate values across observations and (for supervised preparation steps) the empirical association between the covariate values and the values of the target variable.

The interpretation of $\text{CVIIM}_{\boldsymbol{s},n,K}$ and results from real data together with expectations regarding the impact of specific data preparation steps give rise to the following

tentative rules of thumb for categorizing the computed values in terms of the impact of CV incompleteness with regard to the considered step(s): $[0, 0.02] \sim$ no impact, $]0.02, 0.1] \sim$ weak, $]0.1, 0.2] \sim$ medium, $]0.2, 0.4] \sim$ strong, $]0.4, 1] \sim$ very strong.

In the following an artificial example is outlined to demonstrate, step by step, a possible application of $\text{CVIIM}_{P,n,K}$. The interest lies in measuring the extent of overoptimism connected with performing the quantile normalization step of RMA before CV in gene-expression-based classification. Suppose there is a dataset with gene expression measurements from 32 patients suffering from breast cancer and from 22 disease-free patients. For each patient there are measurements of the expression of 54675 genes. As a classification method the nearest shrunken centroids (NSC) are used (Tibshirani et al.; 2002). The error $e_{incompl,5}(\boldsymbol{s})$, as estimated by incomplete 5-fold CV, is computed by conducting RMA normalization beforehand on the entire dataset and performing 5-fold CV on the normalized dataset. In this procedure only the fitting of NSC is repeated in each CV iteration on the training datasets. The CV is repeated 300 times to obtain more stable results. The full CV error $e_{full,5}(\boldsymbol{s})$ is computed by performing a 5-fold CV in which the quantile normalization step of RMA (as well as the fitting of the NSC) is re-performed in each CV iteration on the respective training set, with addon normalization of the corresponding test set through the addon procedure by Kostka and Spang (2008). This procedure is repeated another 300 times. If $e_{incompl,5}(\boldsymbol{s}) = 0.15$ and $e_{full,5}(\boldsymbol{s}) = 0.1503$ were obtained, then $\text{CVIIM}_{\boldsymbol{s},n,K} = 1 - 0.15/0.1503 \sim 0.002$. According to the aforementioned rules of thumb this would indicate no impact on the estimated error.

This result obtained for a specific dataset and specific classifier, may not be representative of all datasets and classifiers in the field of gene-expression-based classification. It is necessary to study several datasets and several analysis settings representative of the considered field in order to formulate recommendations regarding incomplete CV for a particular step. Alternatively, specific guidelines could be formulated for particular settings and data types within the considered field; however, this could result in overly complicated guidelines.

For a detailed formal introduction to the concepts presented in this section such as prediction rules, prediction error, and its estimation via full and incomplete CV, consult Appendices A.2.1 and A.2.2.

2.2.6. Global CVIIM

Clearly, the value of $\text{CVIIM}_{\boldsymbol{s},n,K}$ depends on the specific dataset. For a general assessment of the bias attributable to a specific step a more global measure summarizing the results obtained for several datasets is needed: The *global CVIIM* is defined as

the quantity resulting when replacing $\mathbb{E}[e_{incompl,K}(\boldsymbol{S})]$ and $\mathbb{E}[e_{full,K}(\boldsymbol{S})]$ in Eq. (2.1) by quantities very similar to their means over the universe of datasets from the area of interest (see Boulesteix et al. (2015) for a more formal description of this concept in another context). Consider the following example: At this time the standard approach to microarray data analysis is to perform quantile normalization of RMA on the entire dataset before performing CV. Suppose that the prediction error is, on average, 0.2 over all datasets from the area of interest, but if full CV were performed with respect to quantile normalization it would equal 0.201. The global CVIIM in this scenario would be $1 - 0.2/0.201 \sim 0.005$, a negligible overall bias.

To estimate the global CVIIM the plug-in estimator can be used, which functions by replacing $e_{incompl,K}(\boldsymbol{s})$ and $e_{full,K}(\boldsymbol{s})$ in Eq. (2.2) with the averages of their values obtained for several datasets from the considered area of application:

$$\text{CVIIMglobal}_{\boldsymbol{s}^{(1)},\ldots,\boldsymbol{s}^{(L)};K} := \begin{cases} 1 - \dfrac{\frac{1}{L}\sum_{l=1}^{L} e_{incompl,K}(\boldsymbol{s}^{(l)})}{\frac{1}{L}\sum_{l=1}^{L} e_{full,K}(\boldsymbol{s}^{(l)})} & \text{if } \frac{1}{L}\sum_{l=1}^{L} e_{incompl,K}(\boldsymbol{s}^{(l)}) < \frac{1}{L}\sum_{l=1}^{L} e_{full,K}(\boldsymbol{s}^{(l)}) \\ & \text{and } \frac{1}{L}\sum_{l=1}^{L} e_{full,K}(\boldsymbol{s}^{(l)}) > 0 \\ 0 & \text{otherwise,} \end{cases} \tag{2.3}$$

where $\boldsymbol{s}^{(1)}, \ldots, \boldsymbol{s}^{(L)}$ are the datasets used. Note that this estimator is not affected considerably by individual extreme CVIIM estimates, which can occur in the case of very small values of $\mathbb{E}[e_{full,K}(\boldsymbol{S})]$. For a detailed discussion on this phenomenon, see Appendix A.2.3.

2.2.7. Illustration

To give a preliminary illustration of the application of CVIIM as a proof of concept, it was applied to supervised variable selection, which was expected to yield high CVIIM values. The following datasets were used: `ProstatecTranscr`, `GSE33205`, `GSE20189`, and `GSE37356`. These also were considered in the PCA study; see Table 2.1.

For each variable a two-sample t-test was conducted to test the equality of the means of the two groups. The variables with the smallest p-values were selected. Because the result was expected to depend substantially on the number of selected variables, the analysis was repeated for different numbers of variables: 5, 10, 20, and half of the total number p of variables. After selecting 5, 10, and 20 variables, linear

discriminant analysis (LDA) was used as a classification method. When selecting half of the variables, LDA could not be applied because the empirical covariance matrices involved are not well behaved in general when the number of variables is higher than the number of observations. In this case, diagonal linear discriminant analysis was used, that is, LDA under the simplifying assumption that within the two classes the variables are independent; see Hastie et al. (2009).

In all the analyses performed in this chapter, $e_{incompl,K}(\boldsymbol{s})$ and $e_{full,K}(\boldsymbol{s})$ were obtained by averaging the results from $B = 300$ runs of K-fold CV, where K takes the values 3, 5, and 10 successively.

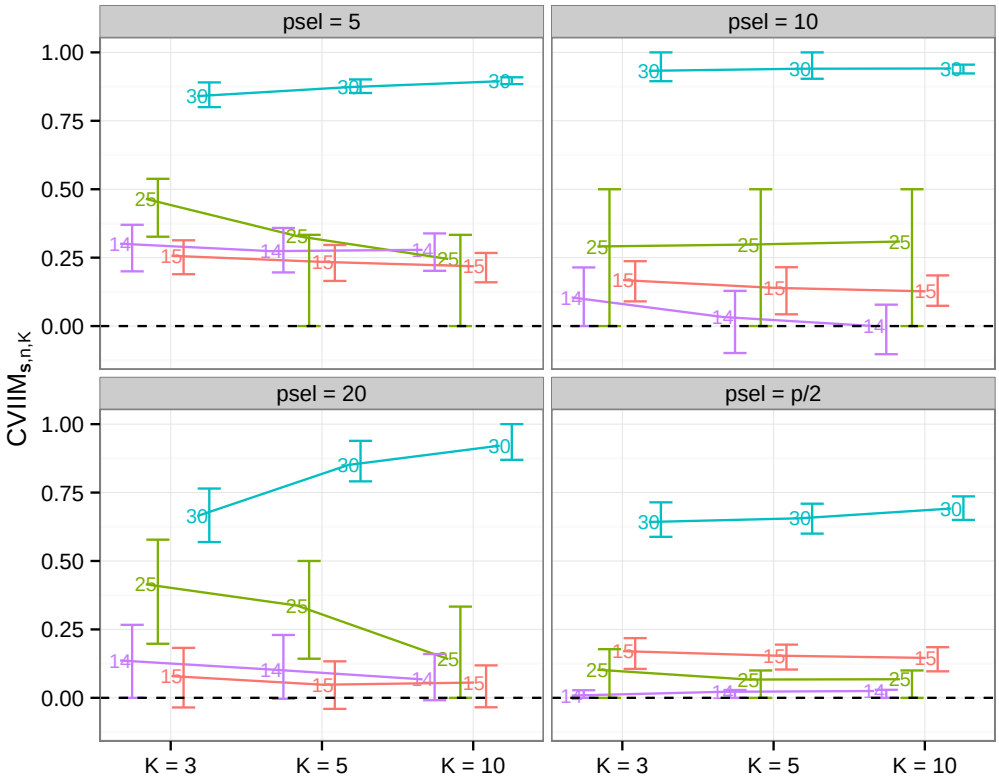

Figure 2.1.: CVIIM$_{\boldsymbol{s},n,K}$ values from variable selection study. The numbers distinguish the datasets. `psel` denotes the number of selected variables.

The CVIIM$_{\boldsymbol{s},n,K}$ values obtained for all settings are displayed in Figure 2.1. In the plots the error bars represent the 25% and 75% quartiles (computed over the $B = 300$ iterations) of the iterationwise non-truncated incompleteness measure estimates (INIMEs) CVIIM$_{\boldsymbol{s},n,K,b} := 1 - e_{incompl,K}(\boldsymbol{s})_b / e_{full,K}(\boldsymbol{s})_b$, where the index b indicates that these errors were obtained for run b (with $b = 1, \ldots, B$). It is important to bear in mind that the error bars should be used for comparisons between each other only, since their absolute lengths have no relevant interpretation. Note that due to

number of sel. variables	$K = 3$	$K = 5$	$K = 10$
5	0.5777	0.5927	0.6126
10	0.5557	0.5617	0.5505
20	0.3971	0.4706	0.4511
p/2	0.2720	0.2702	0.2824

Table 2.2.: Estimates of global CVIIM from variable selection study

the unboundedness of the INIMEs, the error bars—as opposed to the $\text{CVIIM}_{\boldsymbol{s},n,K}$ values—are not bound by zero.

While $\text{CVIIM}_{\boldsymbol{s},n,K}$ is especially large for small numbers of selected variables, relatively large values also are observed when half of the variables are selected (with the exception of the dataset with the fewest variables). Although the differences in $\text{CVIIM}_{\boldsymbol{s},n,K}$ for the selection of 5, 10, and 20 variables are not large, the estimates of the global CVIIM given in Table 2.2 indicate that the bias induced by incomplete CV tends to decrease with an increasing number of selected variables.

Dataset 30 stands out through its noticeably larger $\text{CVIIM}_{\boldsymbol{s},n,K}$ values in all plots. This dataset comprises only 36 observations but 47231 variables (see Table 2.1), which at least may explain in part the larger values. Extreme values above 0.9, however, are surprising.

In this illustrative analysis, employing the new measure CVIIM the following conclusion previously obtained in the literature could be confirmed: Performing supervised variable selection before CV leads to a strong bias of the resulting error estimate.

2.2.8. Study design

The investigation of normalization is based on the first 12 microarray datasets listed in Table 2.1. Here, the two variants of normalization described in section 2.2.3 were used. Two classification methods were employed successively to derive prediction rules: NSC and LDA performed on partial least squares components (PLS-LDA)(Boulesteix; 2004). For NSC the shrinkage intensity Δ was chosen from the grid $\{0.05, 0.1, 0.25, 0.5, 1, 1.5\}$ and for PLS-LDA the number of components n_{comp} was chosen from the grid $\{1, 2, \ldots, 10\}$. Parameter choice was done in the following way. For each considered training dataset, 3-fold internal CV was performed for each candidate parameter value from the grid. The candidate parameter value yielding the smallest 3-fold CV error was selected.

The study of PCA is based on the last 20 microarray datasets listed in Table 2.1. The constructed principal components were used as predictors in LDA and RF, successively. For RF, the crucial parameter *mtry*, denoting the number of predictors

considered as candidates in the splits of the trees, was chosen by 3-fold internal CV from the grid $\{1, 2, 3, 5, 10\}$. Since the results can be assumed to depend heavily on the number of principal components used as predictors, the analyses were repeated for four numbers: 2, 5, 10, and 15.

2.3. Results

2.3.1. Normalization

Figure 2.2 depicts the $\text{CVIIM}_{\boldsymbol{s},n,K}$ values from the normalization study together with the estimates of global CVIIM. The latter are given in Table 2.3.

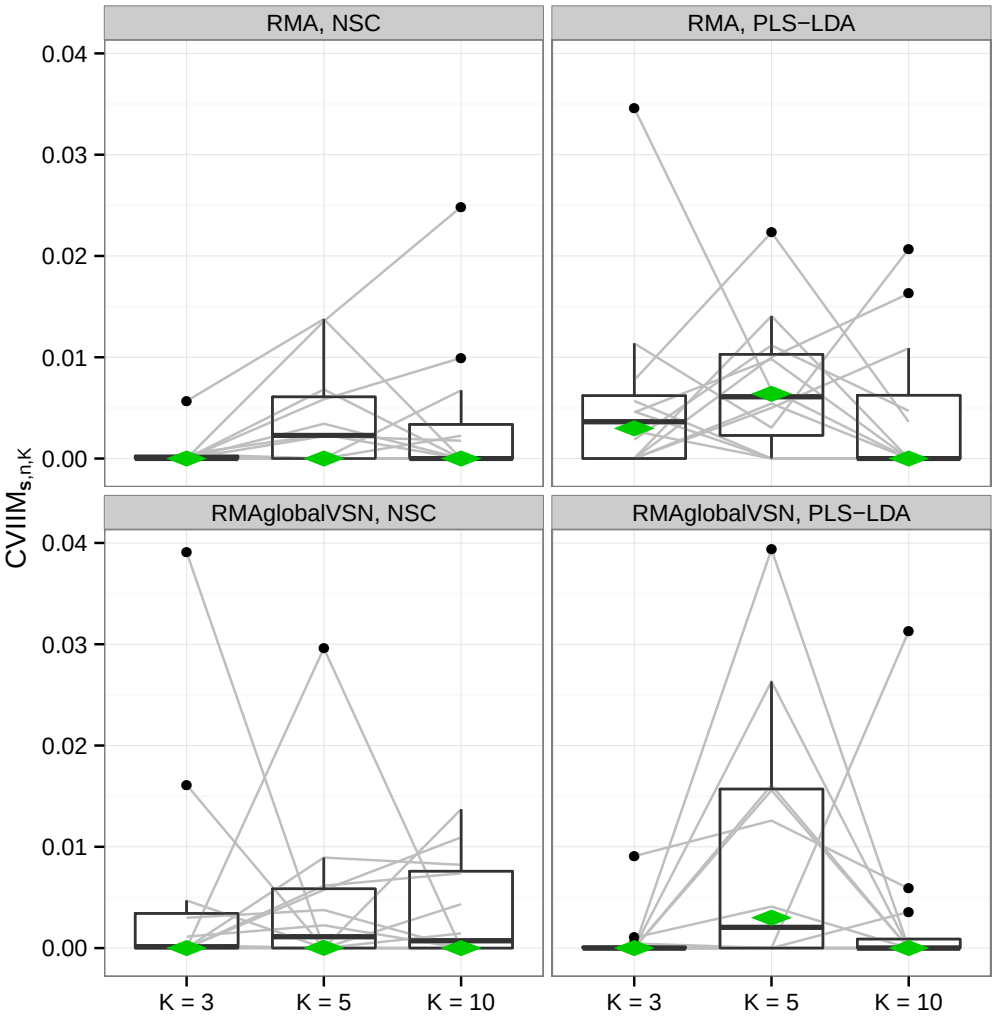

Figure 2.2.: $\text{CVIIM}_{\boldsymbol{s},n,K}$ values from normalization study. The grey lines connect the values corresponding to the same datasets. The diamonds depict the estimates of global CVIIM.

Normalization method	Classification method	$K = 3$	$K = 5$	$K = 10$
RMA	NSC	0.0000	0.0000	0.0000
	PLS-LDA	0.0030	0.0064	0.0000
RMAglobalVSN	NSC	0.0000	< 0.0001	0.0000
	PLS-LDA	0.0000	0.0030	0.0000

Table 2.3.: Estimates of global CVIIM from the normalization study

For both normalization approaches the $\text{CVIIM}_{\boldsymbol{s},n,K}$ values are very small for all datasets and both classifiers. In the majority of cases the measure estimates suggest no bias resulting from incomplete CV for normalization as defined by the rules of thumb given on page 21. The global CVIIM estimates seem to confirm that in general there is no bias. Slightly higher values are obtained for PLS-LDA than for NSC, but the difference is not noteworthy.

For the individual datasets there is no visible dependence of the measure estimates on K, although in general a negative dependence is expected; see section 2.4.5 for a discussion on this topic. The fact that such a decrease with K is not observed for normalization likely can be explained by the small values of the estimates: $e_{incompl,K}(\boldsymbol{s})$ and $e_{full,K}(\boldsymbol{s})$ are very similar here. Therefore, the non-systematic fluctuations across the different K values are attributable to small—probably random—fluctuations of $e_{incompl,K}(\boldsymbol{s})$ and $e_{full,K}(\boldsymbol{s})$ over K, which could overshadow a potential dependence on K.

In contrast to section 2.2.7, no iteration-based error bars for the individual $\text{CVIIM}_{\boldsymbol{s},n,K}$ values are presented here. When depicting the results of a study with a larger number of datasets, individual error bars make the corresponding plots increasingly unclear. In this situation it is possible to focus on the distribution of the $\text{CVIIM}_{\boldsymbol{s},n,K}$ values across datasets; the results for individual datasets are less important. Nevertheless, extreme individual results should be examined more closely.

Given the small CVIIM estimates it can be concluded that RMA and RMAglobalVSN can be performed safely before CV without the danger of inducing a relevant bias in the resulting error estimate.

2.3.2. Principal Component Analysis

Figure 2.3 and Table 2.4 show the results of the PCA study. Note that the scale of Figure 2.3 is much larger than that of the corresponding plot for normalization (Figure 2.2). Globally, the results suggest a weak but existing underestimation of the true error $\mathbb{E}[e_{full,K}(\boldsymbol{S})]$ when performing PCA before CV. Exceptions are LDA in those instances where the number of components is greater than five, where zero

Classification method	number of components	$K = 3$	$K = 5$	$K = 10$
LDA	2	0.0974	0.0805	0.0582
	5	0.0397	0.0371	0.0354
	10	0.0000	0.0000	0.0000
	15	0.0000	0.0000	0.0000
RF	2	0.0855	0.0747	0.0659
	5	0.0686	0.0558	0.0516
	10	0.0907	0.0613	0.0368
	15	0.1117	0.0988	0.0794

Table 2.4.: Estimates of global CVIIM from the PCA study

values of the global CVIIM estimates are obtained.

For LDA the impact of incomplete CV seems to diminish with an increasing number of components in PCA. The global CVIIM estimates generally are larger for RF than for LDA. While the overall effects of performing PCA before CV seem to be weak, Figure 2.3 reveals that there are several settings in which the CVIIM estimates suggest a strong bias, according to the rules of thumb given on page 21, for a non-neglible number of datasets. Therefore, these results strongly favor the use of full CV over incomplete CV with respect to PCA.

A closer look at Table 2.4 reveals that, in general, the global CVIIM estimates decrease with an increasing value of K (for all settings with non-zero values). For example, this decrease is noticeable for LDA with $ncomp = 2$ and RF with $ncomp = 10$. This suggests that the estimates of global CVIIM are overly high in these cases due to the greater upward bias of $e_{full,K}(\boldsymbol{s})$ compared to $e_{incompl,K}(\boldsymbol{s})$ as detailed in section 2.4.5. The global CVIIM estimates depend on the means of the $e_{full,K}(\boldsymbol{s})$ and the $e_{incompl,K}(\boldsymbol{s})$ values calculated over the included datasets. The decrease with larger values of K is induced by the mean of the $e_{full,K}(\boldsymbol{s})$ values becoming more similar to the mean of the $e_{incompl,K}(\boldsymbol{s})$ values with increasing value of K. For most settings there is no substantial decrease in the global CVIIM estimates. This suggests that the two cases for which the decrease with K was strong are connected to aberrant results for individual datasets, which was confirmed by inspecting more closely the individual values obtained for each setting and each dataset.

More precisely, a simple type of sensitivity analysis was performed. First, for each of the two settings that dataset displaying the largest difference between $e_{full,3}(\boldsymbol{s})$ and $e_{full,10}(\boldsymbol{s})$ was omitted and the global CVIIM values were re-estimated. For the LDA with $ncomp = 2$ the results were 0.0812 ($K = 3$), 0.0681 ($K = 5$), and 0.0524 ($K = 10$), and for RF with $ncomp = 10$ the following values were obtained: 0.0590 ($K = 3$), 0.0351 ($K = 5$), and 0.0222 ($K = 10$). The values are obviously more similar across the three different K values for both settings than the results obtained

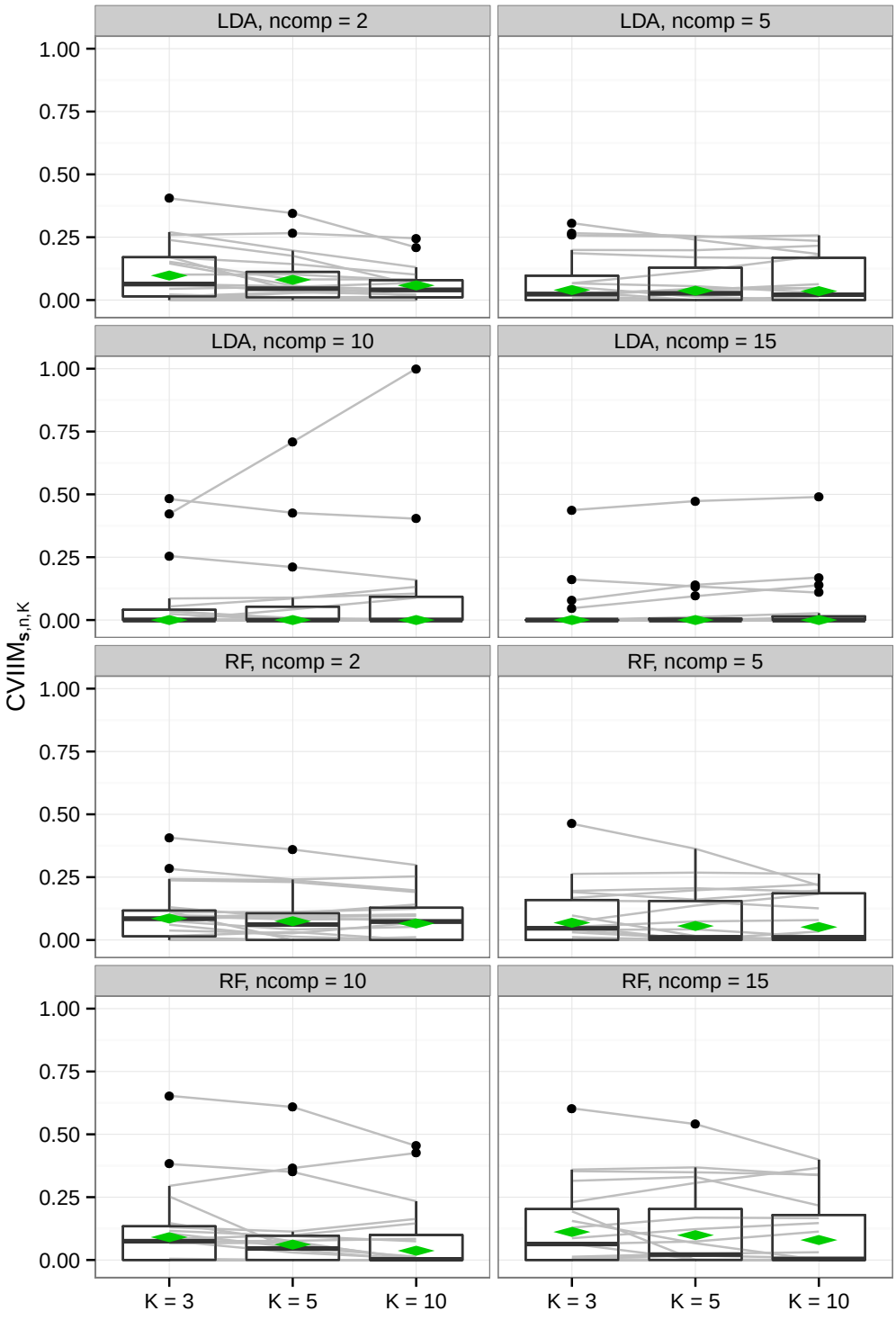

Figure 2.3.: CVIIM$_{\boldsymbol{s},n,K}$ values from PCA study. The grey lines connect the values corresponding to the same datasets. The diamonds depict the estimates of global CVIIM.

when using all 20 datasets; see Table 2.4. This is especially noticeable in the case of the values for $K = 5$ and $K = 10$ in "LDA with $ncomp = 2$"; nevertheless, there still are substantial differences. Therefore, as a second step the same procedure was repeated. However, this time the three datasets with the largest differences between $e_{full,3}(\boldsymbol{s})$ and $e_{full,10}(\boldsymbol{s})$ were omitted. The results were as follows: 0.0676 ($K = 3$), 0.0575 ($K = 5$), and 0.0499 ($K = 10$) for LDA with $ncomp = 2$, and 0.0067 ($K = 3$), 0.0000 ($K = 5$), and 0.0000 ($K = 10$) for RF with $ncomp = 10$. For the former setting the similarity across K values obviously has increased, but the size of the values has not decreased very much. The (almost) zero values for the second setting are quite striking given that values as high as 0.0907 for $K = 3$ were observed when using all 20 datasets. The same analysis also was performed for all other settings (results not shown): The global CVIIM estimates in these settings tended to be more robust to the removal of datasets than the ones of the settings presented here. Such results, especially those obtained for the setting "RF with $ncomp = 10$", illustrate that a large decrease in global CVIIM estimates with an increasing value of K should be interpreted with caution. In such cases it is recommendable to conduct a sensitivity analysis of the same kind as the one conducted here.

2.4. Discussion

In the following, first, possible alternative measures of CV incompleteness are examined and why they are less appropriate than the new measure CVIIM is discussed. Then, as an outlook some preliminary results obtained for additional data preparation steps beyond normalization and PCA are presented. Finally, various further issues related to CVIIM are explored.

2.4.1. Alternative measures of CV incompleteness

An important question with respect to the definition of CVIIM is whether it depends on $\mathbb{E}[e_{full,K}(\boldsymbol{S})]$. Such dependence is undesirable because CVIIM should not be a measure of the error but rather of the impact of CV incompleteness. To examine this in the context of the PCA study, the upper panel of Figure 2.4 shows a plot of the $\text{CVIIM}_{\boldsymbol{s},n,K}$- against the $e_{full,K}(\boldsymbol{s})$ values, where the different analysis settings for a given dataset are represented using the same color and number, and the mean over the values for each dataset is represented with a black point. This plot suggests no relevant dependence of $\text{CVIIM}_{\boldsymbol{s},n,K}$ on the full CV error $e_{full,K}(\boldsymbol{s})$. For two of the smallest errors, extreme CVIIM estimates resulting from random fluctuations in the error estimates as discussed in Appendix A.2.3 can be observed. However, this

problem—concerning only two values of 480 error values in total—seems to be negligible. The lower panel of Figure 2.4 displays the zero-truncated difference between $e_{full,K}(\boldsymbol{s})$ and $e_{incompl,K}(\boldsymbol{s})$ against $e_{full,K}(\boldsymbol{s})$. This plot clearly suggests a comparatively strong dependence of the estimates of this measure on the full CV error, as also observed in the results obtained in the simulation study presented in Appendix A.1, and thus provides evidence supporting the use of a ratio-based measure rather than a difference-based measure. Analogous plots give a very similar picture in the case of normalization; see Figure 2.5.

An obvious, but less insightful, way of visualizing the impact of CV incompleteness is simply to plot $e_{full,K}(\boldsymbol{s})$ and $e_{incompl,K}(\boldsymbol{s})$ for the individual datasets. Figure 2.6 shows such a plot for the PCA study. Without closer inspection, it appears that in some cases $e_{incompl,K}(\boldsymbol{s})$ is considerably smaller than $e_{full,K}(\boldsymbol{s})$, indicating the strong bias already suggested by the $\text{CVIIM}_{\boldsymbol{s},n,K}$ values.

However, this visualization has two crucial disadvantages. First, in contrast to the plot of the CVIIM estimates, it does not show values which allow immediate interpretation of the extent of overoptimism for the individual datasets. Second, it draws attention to the different sizes of the errors across individual datasets rather than highlight the discrepancies between the $e_{full,K}(\boldsymbol{s})$ and $e_{incompl,K}(\boldsymbol{s})$ values, which should be the focus.

2.4.2. Outlook: other preparation steps

Additional analyses were performed for additional data preparation steps, however, with fewer datasets and fewer analysis settings than in the studies for normalization and PCA. These preparation steps included optimization of tuning parameters, variable filtering by variance, and imputation of missing values. In the following, the study designs and detailed results of these analyses are described.

Optimization of tuning parameters An important data preparation step is choosing tuning parameter values. Seven classification methods were considered successively, each with one tuning parameter of interest optimized from a grid through internal CV as described in section 2.2.8: the number of iterations m_{stop} in componentwise boosting with logistic loss function (LogitBoost) (grid: $\{50, 100, 200, 500, 1000\}$) (Bühlmann and Yu; 2003), the number of neighbors in the k-Nearest Neighbors algorithm (kNN) (grid: $\{1, 2, \ldots, 10\}$), the shrinkage intensity in L_1-penalized logistic regression expressed as the fraction of the coefficient L_1-norm compared to the maximum possible L_1-norm (grid: $\{0.1, 0.2, \ldots, 0.9\}$) (Young-Park and Hastie; 2007), the shrinkage intensity for the class centroids in NSC (grid: $\{0.1, 0.25, 0.5, 1, 2, 5\}$), the

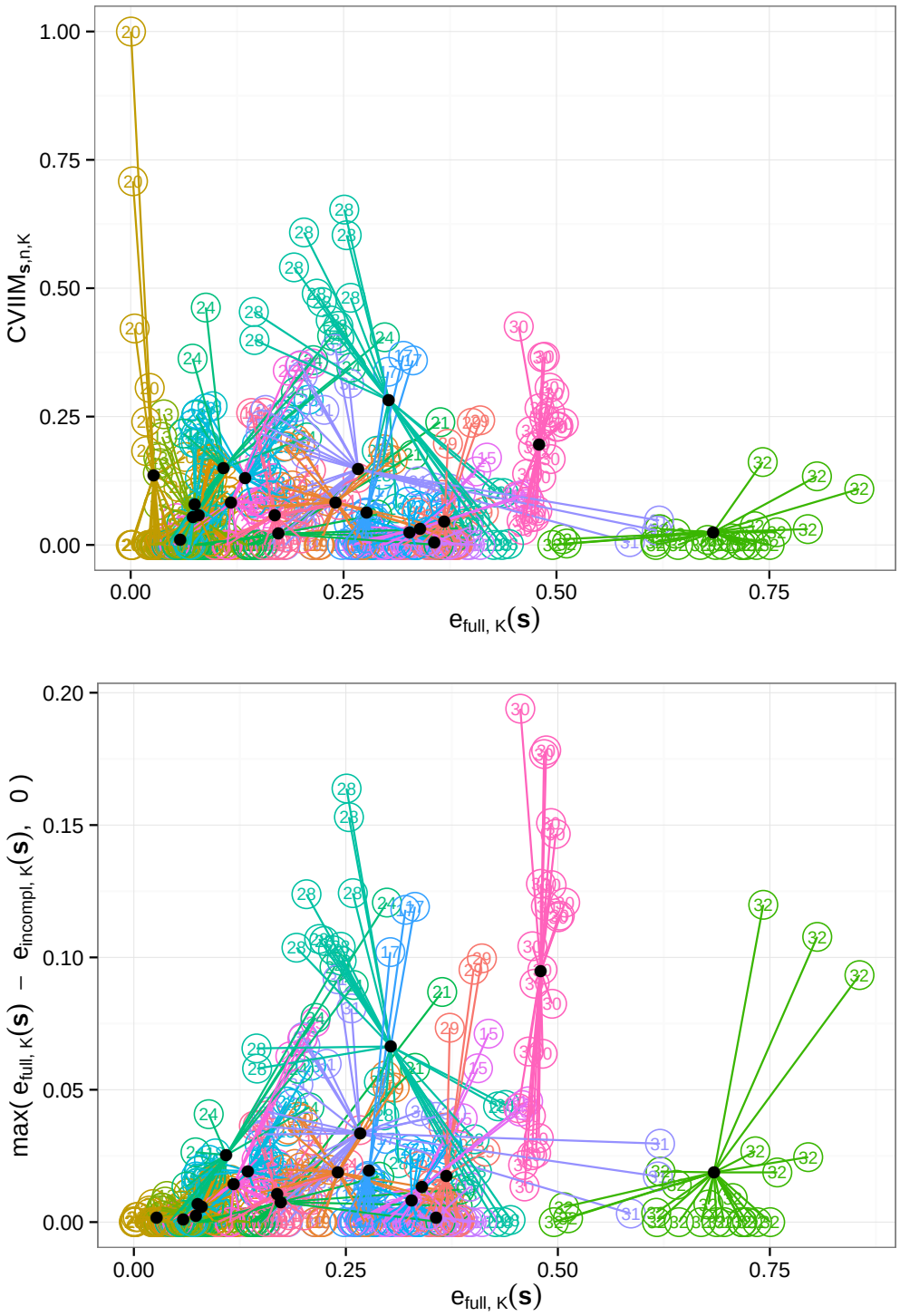

Figure 2.4.: Dependence on CV errors in PCA study. Upper panel: $\text{CVIIM}_{\boldsymbol{s},n,K}$ values versus $e_{full,K}(\boldsymbol{s})$ values for all settings; Lower panel: Zero-truncated differences of $e_{full,K}(\boldsymbol{s})$ values and $e_{incompl,K}(\boldsymbol{s})$ values against $e_{full,K}(\boldsymbol{s})$ values for all settings. The colors and numbers distinguish the different datasets. The filled black circles depict the respective means over the results of all settings obtained for the specific datasets.

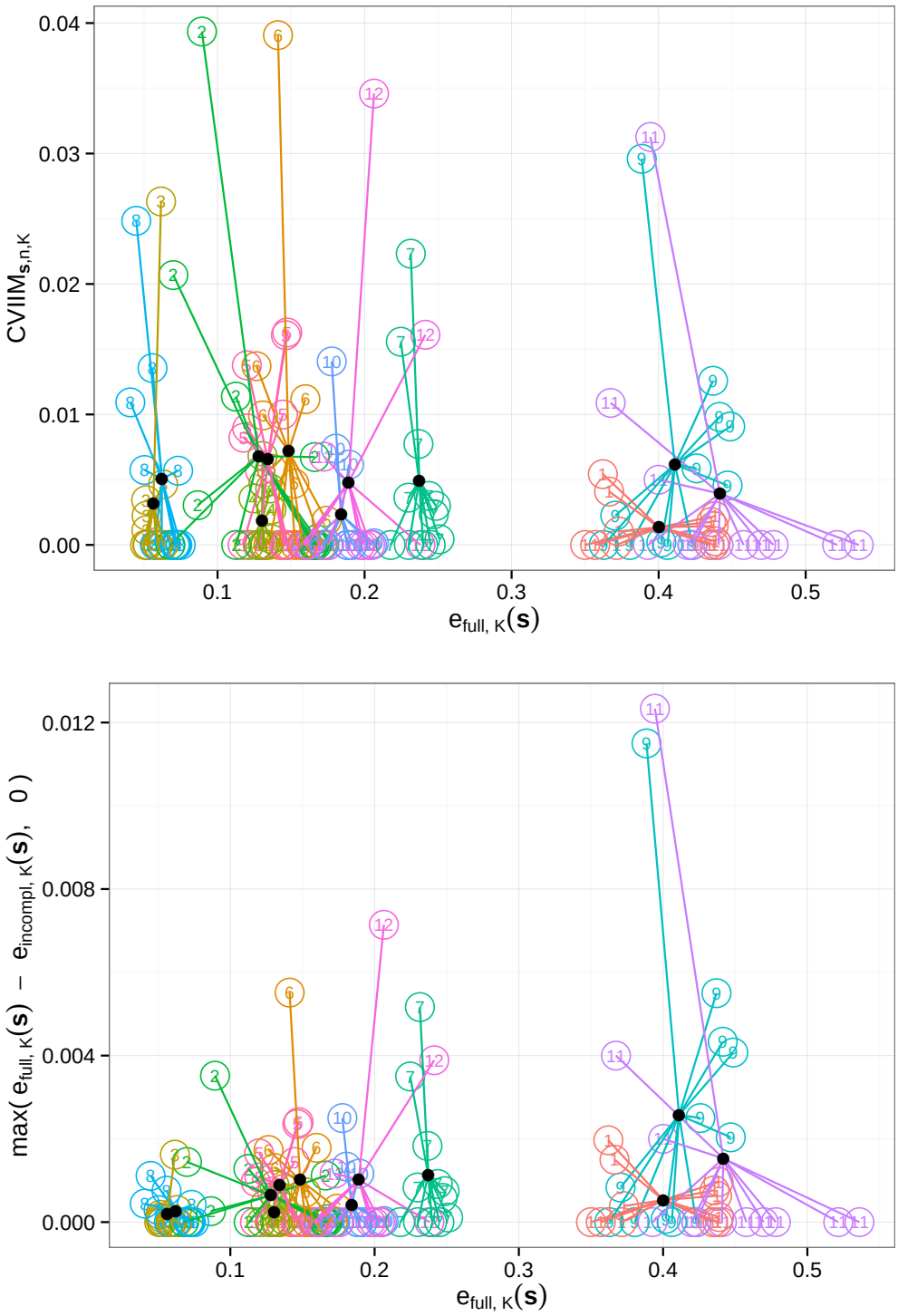

Figure 2.5.: Dependence on CV errors in normalization study. Upper panel: $\text{CVIIM}_{\boldsymbol{s},n,K}$ values against $e_{full,K}(\boldsymbol{s})$ values for all settings; Lower panel: Zero-truncated differences of $e_{full,K}(\boldsymbol{s})$ values and $e_{incompl,K}(\boldsymbol{s})$ values against $e_{full,K}(\boldsymbol{s})$ values for all settings. The colors and numbers distinguish the datasets. The filled black circles depict the respective means over the results of all settings obtained for the specific datasets.

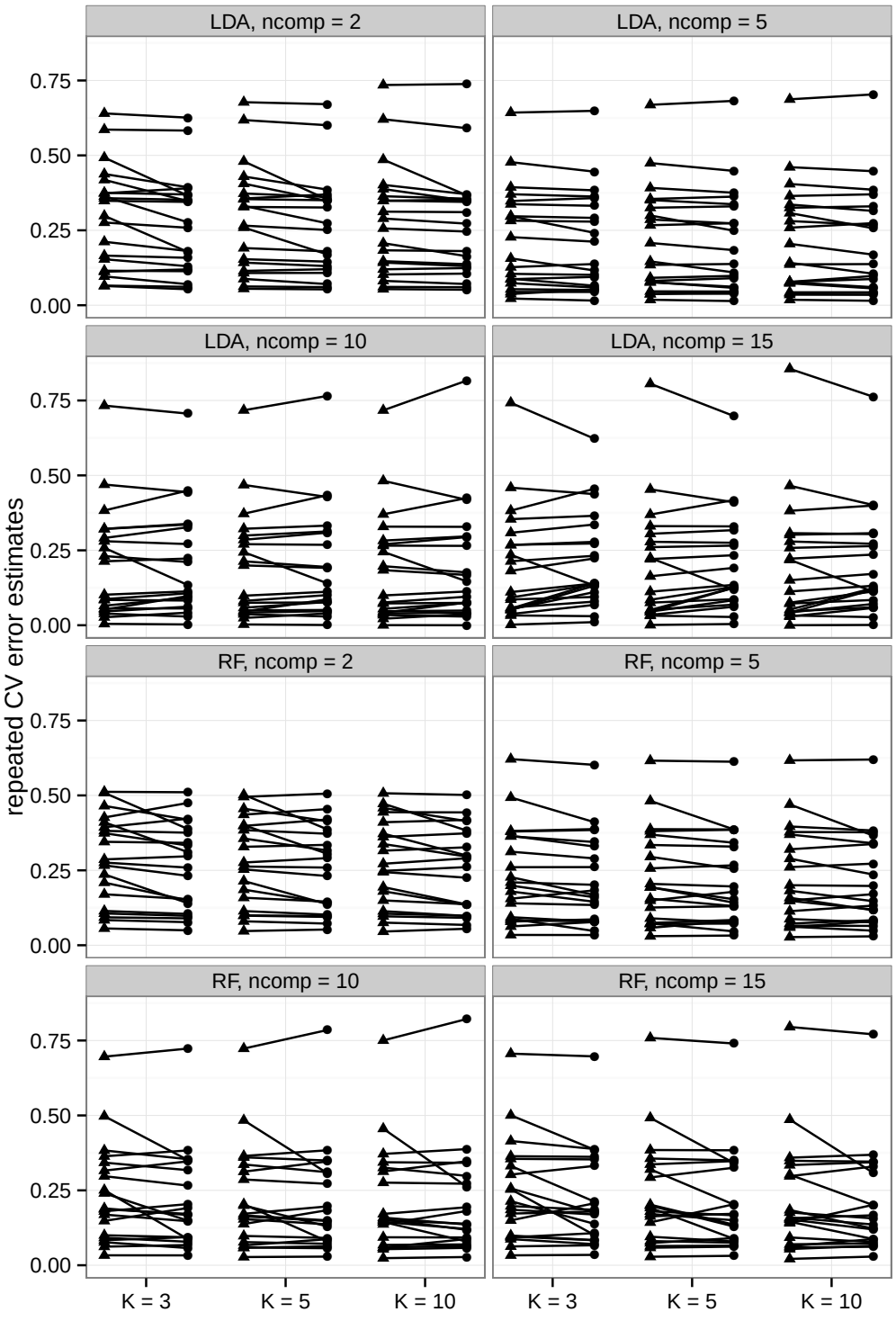

Figure 2.6.: Errors in PCA study. $e_{full,K}(\boldsymbol{s})$ values (filled triangles) and $e_{incompl,K}(\boldsymbol{s})$ values (filled circles) for all datasets and settings from the PCA study

.

number of components in PLS-LDA (grid: $\{1, 2, \ldots, 10\}$) (Boulesteix and Strimmer; 2007), the number *mtry* of variables randomly sampled as candidates at each split in RF (grid: $\{1, 5, 10, 50, 100, 500\}$) (Breiman; 2001), and the cost of constraints violation in support vector machines (SVMs) with linear kernel (grid: $\{10^{-5} \cdot 40^{k/7} : k = 0, \ldots, 7\}$) (Schölkopf and Smola; 2002).

The same four example datasets were used as in the illustrative application of CVIIM to supervised variable selection presented in section 2.2.7. Figure 2.7 and Table 2.5 show the results. None of the methods exhibit large $\text{CVIIM}_{\boldsymbol{s},n,K}$ values. According to the rules of thumb given on page 21, only one result is classified as a medium effect, namely tuning of the number of components in PLS-LDA for dataset `GSE33205` and the rest are classified as weak effects.

Researchers applying CV to optimize tuning parameters in the way described here may be tempted to use as an error estimate of the prediction rule the CV error estimate for the ultimately chosen tuning parameter value obtained during optimization, that is, the smallest one. The optimistic bias of this estimate has been studied in the literature (Varma and Simon; 2006; Bernau et al.; 2013; Boulesteix and Strobl; 2009). Given a large enough number of repetitions of the CV used in the optimization, this optimistic bias becomes equivalent to the bias resulting from performing the optimization before CV, as studied here. This is because the dependence of the CV error estimates, that is, those of the optimization process, on the specific training/test set divisions diminishes with an increasing number of repetitions. As a result, the additional distortion of the estimate studied by Varma and Simon (2006) and Bernau et al. (2013), which was due to the impact of optimally selecting the smallest error estimate, and the distortion of the incomplete CV estimate studied here, decrease in the same way.

An effort was made to choose reasonable parameter grids in the above analysis; however, the results are likely grid-dependent. Moreover, for methods such as RF involving several important tuning parameters, it would be interesting to investigate the bias induced by incomplete CV when optimizing two or more tuning parameters simultaneously.

Variable filtering by variance Here, for each variable the empirical variance was calculated and half of the variables with the largest variances were selected. Such procedures are common in the context of gene expression data analysis. The aim of these procedures is to eliminate genes exhibiting little variation in their profile and therefore generally are not of interest (Soreq et al.; 2012). Again, diagonal linear discriminant analysis was used as a classification method and the same four example datasets were considered as in the case of supervised variable selection and tuning.

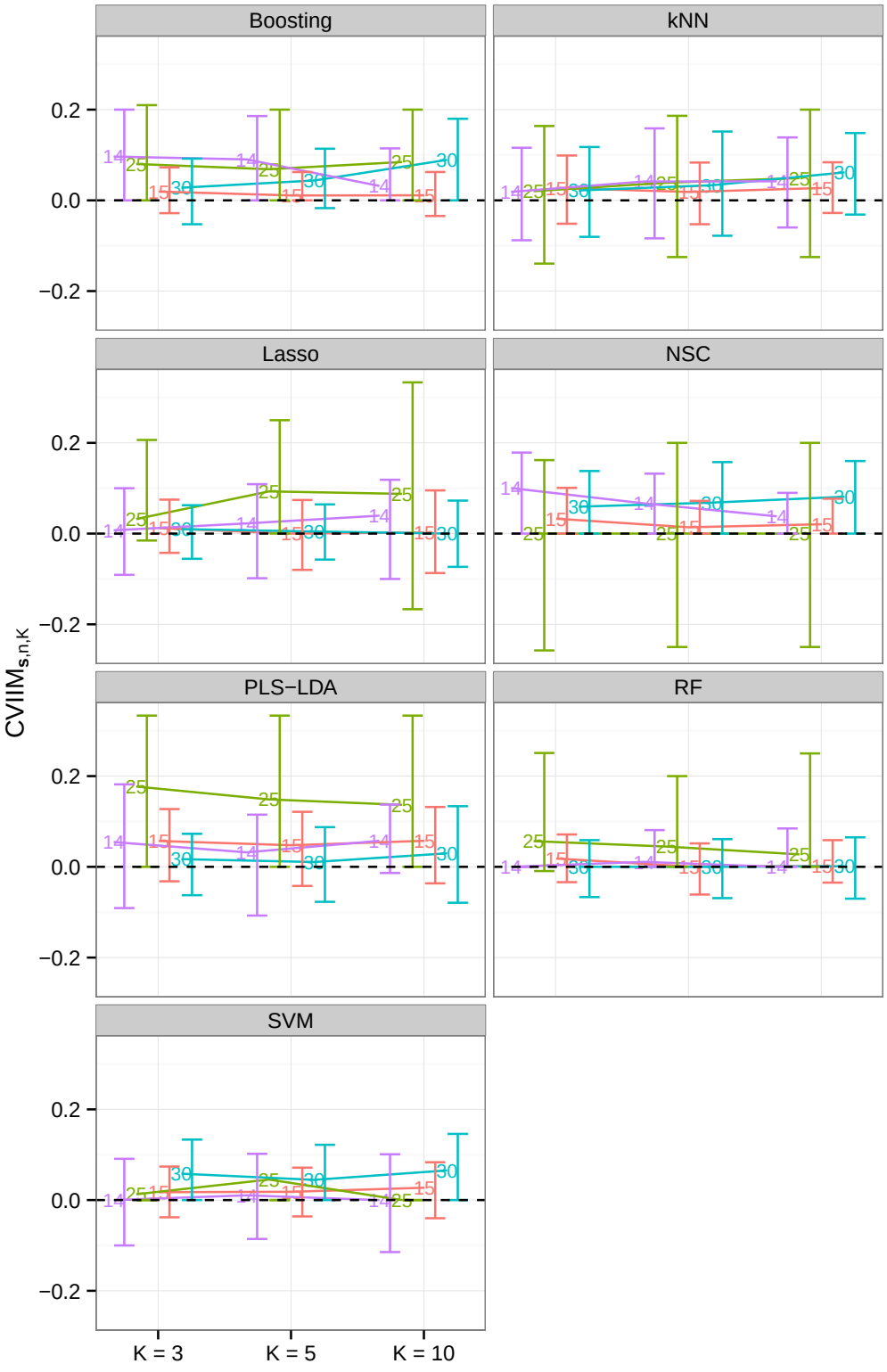

Figure 2.7.: CVIIM$_{s,n,K}$ values from tuning study. The numbers distinguish the datasets.

number of sel. variables	$K = 3$	$K = 5$	$K = 10$
Boosting	0.0385	0.0415	0.0591
kNN	0.0227	0.0309	0.0465
Lasso	0.0124	0.0134	0.0126
NSC	0.0554	0.0472	0.0501
PLS-LDA	0.0456	0.0343	0.0486
RF	0.0097	0.0014	0.0025
SVM	0.0350	0.0330	0.0417

Table 2.5.: Estimates of global CVIIM from the tuning study

Here, only zero or almost zero values are observed, see Figure 2.8. The global CVIIM estimates are correspondingly zero or (for $K = 3$) almost zero. These results suggest that the selection of a large number of variables in an unsupervised fashion might be performed outside CV in contrast to supervised variable selection.

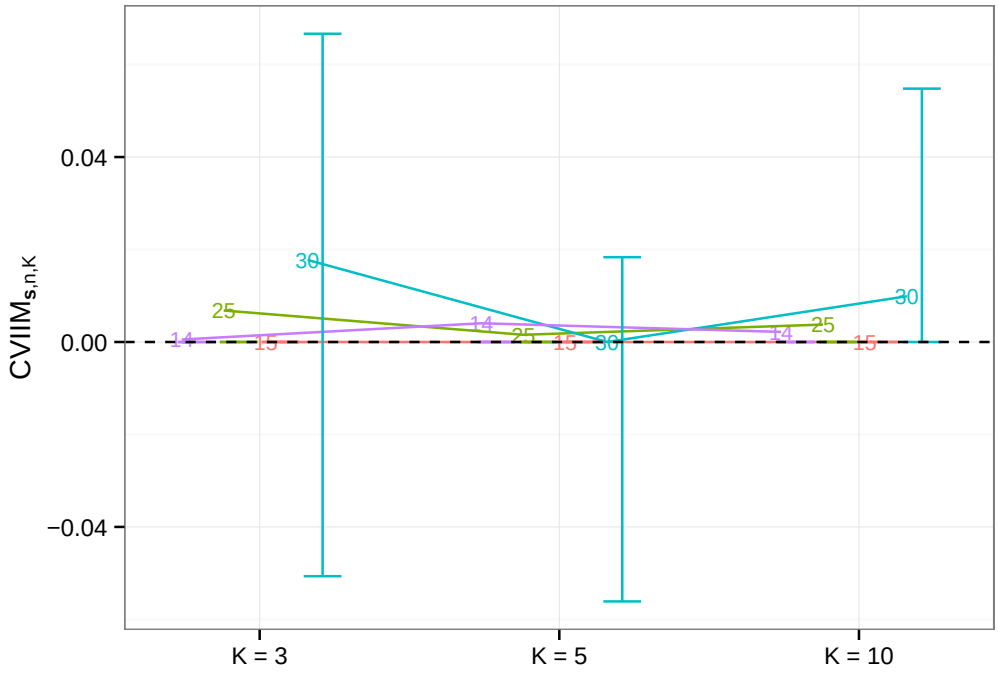

Figure 2.8.: CVIIM$_{\boldsymbol{s},n,K}$ values from variable filtering study. The numbers distinguish the datasets.

Imputation of missing values The k-Nearest Neighbors imputation procedure (Wong; 2013) was applied. The latter commonly is used for the analysis of high-dimensional microarray data. Prior to imputation, the variables were centered and scaled, and the estimated means and standard deviations were stored. After imputing the values, they were rescaled using the stored standard deviations, and the stored

means were added to retransform the data to the original level. The result of the imputation may depend heavily on the number k of nearest neighbors considered. Therefore, to optimize this parameter on the grid $\{1, 2, 3, 5, 10, 15\}$ again 3-fold CV was employed. For correct addon imputation, the means and standard deviations estimated from the training data were used and only the training data were examined when searching for the k nearest neighbors.

First a collection of five example datasets was considered, `GenitInfCow`, containing measurements on 51 cows, 36 of which were suffering from a major genital infection. Each dataset (IDs 33 - 37 in Table 2.1) contains measurements from a specific week and comprises between 21 and 27 variables. These datasets had been obtained with great thanks through personal communication with Michael Schmaußer. RF was used as a classification method here. The second example dataset used was `ProstatecMethyl` (ID 38 in Table 2.1), containing 222 variables obtained through DNA methylation profiling of 70 patients, 29 of whom were suffering from metastatic prostate cancer. This dataset was obtained with great thanks from Thomas Stadler, the fifth author of Hornung et al. (2015). NSC was used as a classification method for this dataset. The shrinkage intensity (for NSC) and *mtry* (for RF) were chosen by 3-fold CV.

The results, shown in Figure 2.9, suggest that in this setting it is irrelevant whether the considered imputation procedure is trained on the entire dataset or based on the training datasets only. The `GenitInfCow` datasets contain proportions of missing values between $\sim 8\%$ and $\sim 19\%$ with tendentiously lower proportions for more advanced weeks. This pattern also is reflected by the $\text{CVIIM}_{\boldsymbol{s},n,K}$ values, where increasingly smaller values are observed for more advanced weeks, with the highest values being observed for Dataset 33, that with the greatest number of missing values. The high-dimensional dataset `ProstatecMethyl` yields $\text{CVIIM}_{\boldsymbol{s},n,K}$ values of zero for all K values. In this dataset only $\sim 3\%$ of values were missing, which is, although small compared to the `GenitInfCow` datasets, a proportion within the range of proportions likely to occur in practice.

2.4.3. Simulation study

In addition to the real data studies presented above, a simulation study was conducted to investigate general statistical properties of $\text{CVIIM}_{\boldsymbol{s},n,K}$. Here, supervised variable selection was used as a preparation step, which displayed the largest $\text{CVIIM}_{\boldsymbol{s},n,K}$ values in the real data analyses. The data-driven simulation study uses the `ProstatecTranscr` dataset and involves 2000 correlated normally distributed predictors. The methods and detailed results are presented in the Appendix A.1.

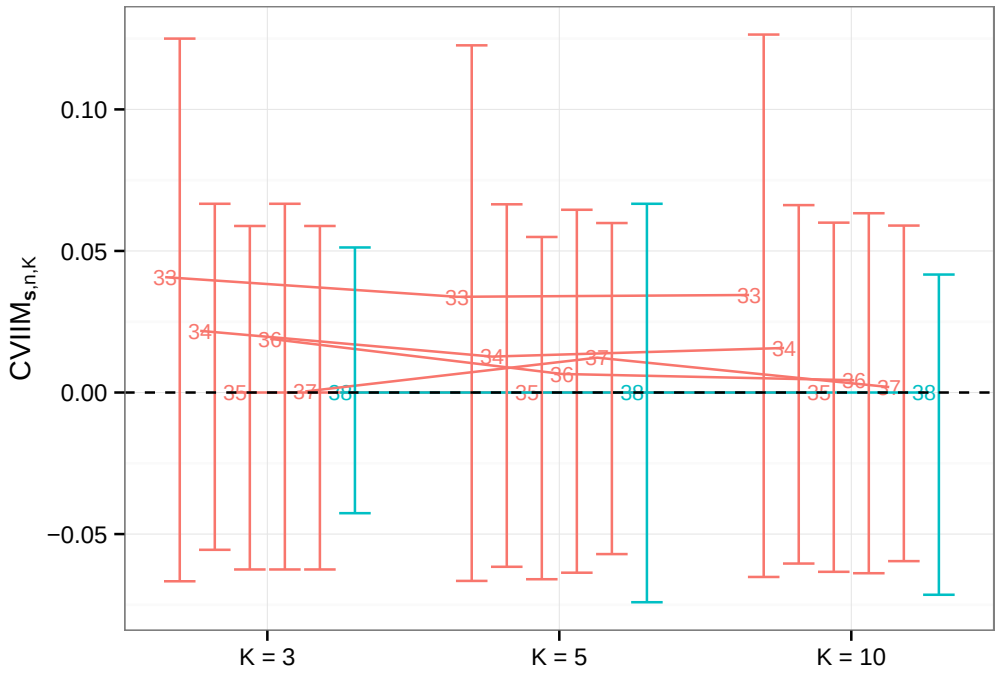

Figure 2.9.: CVIIM$_{\boldsymbol{s},n,K}$ values from imputation study. The numbers distinguish the datasets.

Briefly, in the simulations the variance of CVIIM$_{\boldsymbol{s},n,K}$ as an estimator of CVIIM$_{P,n,K}$ was relatively high and decreased with decreasing CVIIM$_{P,n,K}$ values. The bias was negligible. When displaying the CVIIM$_{\boldsymbol{s},n,K}$ values graphically in sections 2.2.7 and 2.4.2 error bars were added representing the variability in the (untruncated) CVIIM$_{P,n,K}$ estimates from individual repetitions of CV. The assumption made there that this variability measure also reflects the actual variance of CVIIM$_{\boldsymbol{s},n,K}$ was confirmed by the simulation, whereby this similarity in behavior was most pronounced for $K = 3$. This indicates that the error bars obtained for the small K values—of all considered values of K (see section 2.4.5)—are the most appropriate for comparing the variability in individual CVIIM$_{\boldsymbol{s},n,K}$ values.

2.4.4. Combination of several steps

In practice, data preparation frequently involves a combination of several preliminary steps, often performed in a natural order. For example, normalization of microarray data has to be performed before variable selection. However, there also are cases with no predefined order. For example, dichotomization can be conducted before or after variable selection. Given a specific order of the steps, if one step is performed during CV, for obvious technical reasons one also has to perform all subsequent steps

during CV. Of course it also is possible to compute CVIIM globally for the entire combination of steps. In the following an example of such a combination is presented.

The following combination of steps was considered: imputation of missing values, supervised variable selection, and optimization of tuning parameters. For imputation the algorithm described in section 2.4.2 was used. For supervised variable selection the 10 variables with the smallest p-values from Wilcoxon's two-sample tests were chosen. As a classification method RF was used with *mtry* as a tuning parameter. Here, *mtry* was optimized from the grid $\{1, 2, \ldots, 5\}$ through internal CV as described in section 2.2.8.

In addition to $e_{full,K}(\boldsymbol{s})$ (where all three preparation steps are conducted on the training datasets and addon procedures are used to prepare the test datasets) and $e_{incompl,K}(\boldsymbol{s})$ (where all three steps are conducted on the entire dataset), for each step the *step-specific incomplete CV error* (stspincCVerr) was calculated. The value of this measure is obtained when the considered step is performed using the entire dataset and the others are performed within CV. StspincCVerr values are calculated to investigate the contribution of the individual steps to the discrepancy between $e_{full,K}(\boldsymbol{s})$ and $e_{incompl,K}(\boldsymbol{s})$.

In the following the procedures performed to obtain the stspincCVerrs are described. For the first step (imputation), the stspincCVerrs simply were computed by performing the considered step using the entire dataset and then starting the CV repetitions, that is, all subsequent steps were included in the repeated CV. For steps not performed when starting data preparation, different actions must be taken to obtain a stspincCVerr. In fact, each preceding step changes the data. Therefore, it is not feasible to employ ordinary CV in the process of calculating the stspincCVerrs because it is not possible to perform the considered step only once on the entire dataset. Instead, to obtain a stspincCVerr value in these cases, the following CV-like procedure was applied, using a random partition of the dataset into K equally sized folds as in ordinary CV. For $k \in \{1, \ldots, K\}$ the following steps were taken: 1) Perform all steps up to the considered step on the k-th training set, each time adjusting the corresponding test set with the appropriate addon procedure; 2) Merge the training set and test set and perform the considered step on the resulting concatenated dataset; 3) Split the result from 2) into the division of training set and test data again; 4) Perform all remaining data preparation steps on the training set, again, each time adjusting the test set with the appropriate addon procedure; and 5) Calculate the misclassification error of the prediction rule on the test set. Note that the only goal of this procedure is to assess the impact of the individual steps within the combinations. It has no other meaningful application in error estimation in practice.

As in the analyses of single steps, here also $B = 300$ repetitions of the CV(-like) procedures were performed. The dataset `ProstatecMethyl` was used.

The $\text{CVIIM}_{\boldsymbol{s},n,K}$ values are 0.2846, 0.3121, and 0.3442 for $K = 3$, $K = 5$, and $K = 10$, respectively. When calculating the stspincCVerr values (Figure 2.10) it is interesting to notice that when performing only supervised variable selection on the entire dataset the error is virtually identical as when performing all steps on the entire dataset. Correspondingly, incomplete CV based on each of the other steps alone yields results almost no different to those from full CV. Therefore in this example the supervised variable selection is almost completely responsible alone for the difference between the correct and incorrect CV procedures.

In summary, individual influential steps can play a dominating role within combinations of several steps; however, the analysis presented in this section should be considered an illustration. The results described here cannot be generalized because they can depend heavily on the specific setting and dataset used.

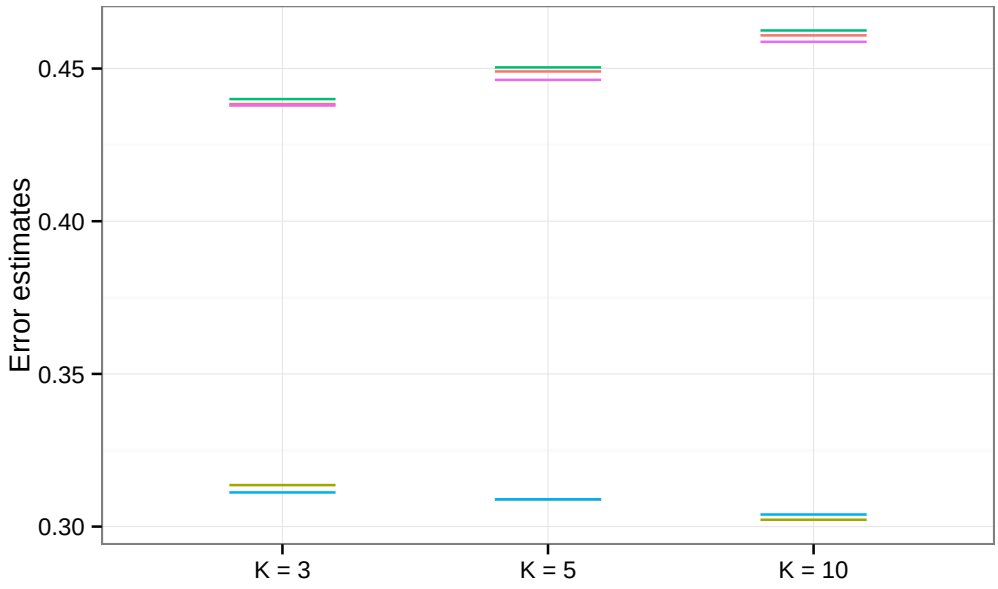

Figure 2.10.: $e_{full,K}(\boldsymbol{s})$ (—) and $e_{incompl,K}(\boldsymbol{s})$ values (—) as well as stspincCVerr values for imputation (—), variable selection (—), and tuning (—).

2.4.5. Further issues

For producing the results presented in section 2.4.2 a limited number of datasets were used in the analyses and therefore these results should not be over-interpreted. In contrast, the results from the normalization and PCA analyses were based on 12 and 20 datasets respectively, and thus are more reliable. As a rule of thumb at least

10 datasets should be analyzed to evaluate the impact of CV incompleteness for a particular preparation step. However, the number of datasets to consider depends on the heterogeneity of the datasets. Generally, variability in the relative performances of different classification methods across different datasets has been found to be large in previous studies (Boulesteix et al.; 2015; Boulesteix; 2013). Frequently analogous observations can be made with respect to the distribution of the CVIIM estimates over datasets. When studying these distributions, implicit variability inherent in individual CVIIM estimates also is observable. This variability probably is difficult to estimate given that the estimator involves a fraction of two CV estimates, the variance of which is very difficult to estimate (Bengio and Grandvalet; 2004).

In CV the training sets are necessarily smaller than the entire dataset and the CV error estimate is, thus, an upwardly biased estimator of the error of the prediction rule fit on the entire dataset. This type of bias also affects the relationship between $\mathbb{E}[e_{full,K}(\boldsymbol{S})]$ and $\mathbb{E}[e_{incompl,K}(\boldsymbol{S})]$. Since in $\mathbb{E}[e_{incompl,K}(\boldsymbol{S})]$ the considered analysis step(s) is/are performed on the entire dataset, the corresponding parameters are estimated more accurately than in $\mathbb{E}[e_{full,K}(\boldsymbol{S})]$ due to the difference in sample sizes. This leads to a greater upward bias of $e_{full,K}(\boldsymbol{s})$ compared to $e_{incompl,K}(\boldsymbol{s})$ with respect to the prediction error of the prediction rule fit on the entire dataset. Occasionally, this can result in increased $\text{CVIIM}_{\boldsymbol{s},n,K}$ values. A strong decrease in the CVIIM estimates with increasing value of K is an indication of the presence of this problem. This is because as the value of K increases, the size of the training sets gets closer to the full sample size, thereby diminishing the additional upward inherent bias of $e_{full,K}(\boldsymbol{s})$. For the latter, see Appendix A.2.1. In most of the analyses performed above no substantial dependence on K was observable. Nevertheless, CVIIM should be estimated for several values of K as a form of sensitivity analysis, as done in the analyses presented in this chapter.

For larger datasets the result of any preliminary step is expected to be more stable, and in fact results become deterministic as the sample size tends to infinity. Therefore, with larger sample sizes the result of a preliminary step will differ less, depending on whether it is performed on the entire dataset or correctly, separating training and test data. Thus CVIIM depends negatively on the sample size. In Figures 2.11, 2.12, and 2.13 for each preparation step investigated the dataset-specific means of the CVIIM estimates over all respective settings are plotted against the sample sizes of the datasets. Here, such a dependence is observable: For large datasets ($n \sim 100$) the CVIIM estimates were much smaller in most cases. This also was observed in the simulation study.

CVIIM in its current form is applicable to binary classification problems only. However, it can be easily adjusted to many other regression problems by replacing

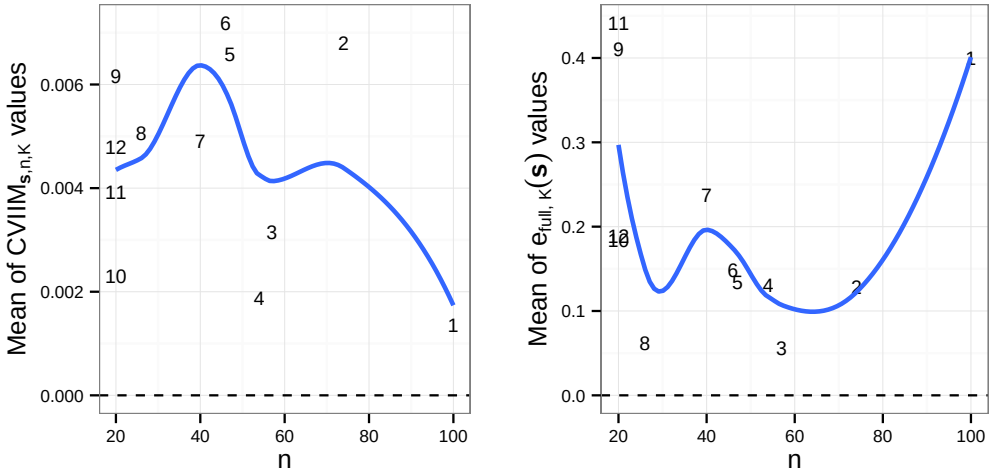

Figure 2.11.: Normalization study: Dataset-specific means of the $\text{CVIIM}_{\boldsymbol{s},n,K}$ and $e_{full,K}(\boldsymbol{s})$ values plotted against the sample sizes of the datasets. The solid lines are LOESS curves.

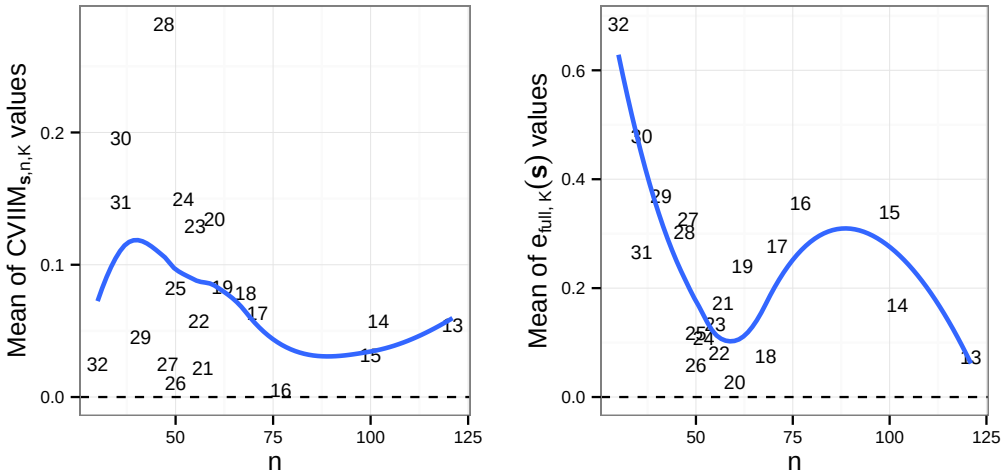

Figure 2.12.: PCA study: Dataset-specific means of the $\text{CVIIM}_{\boldsymbol{s},n,K}$ and $e_{full,K}(\boldsymbol{s})$ values plotted against the sample sizes of the datasets. The solid lines are LOESS curves.

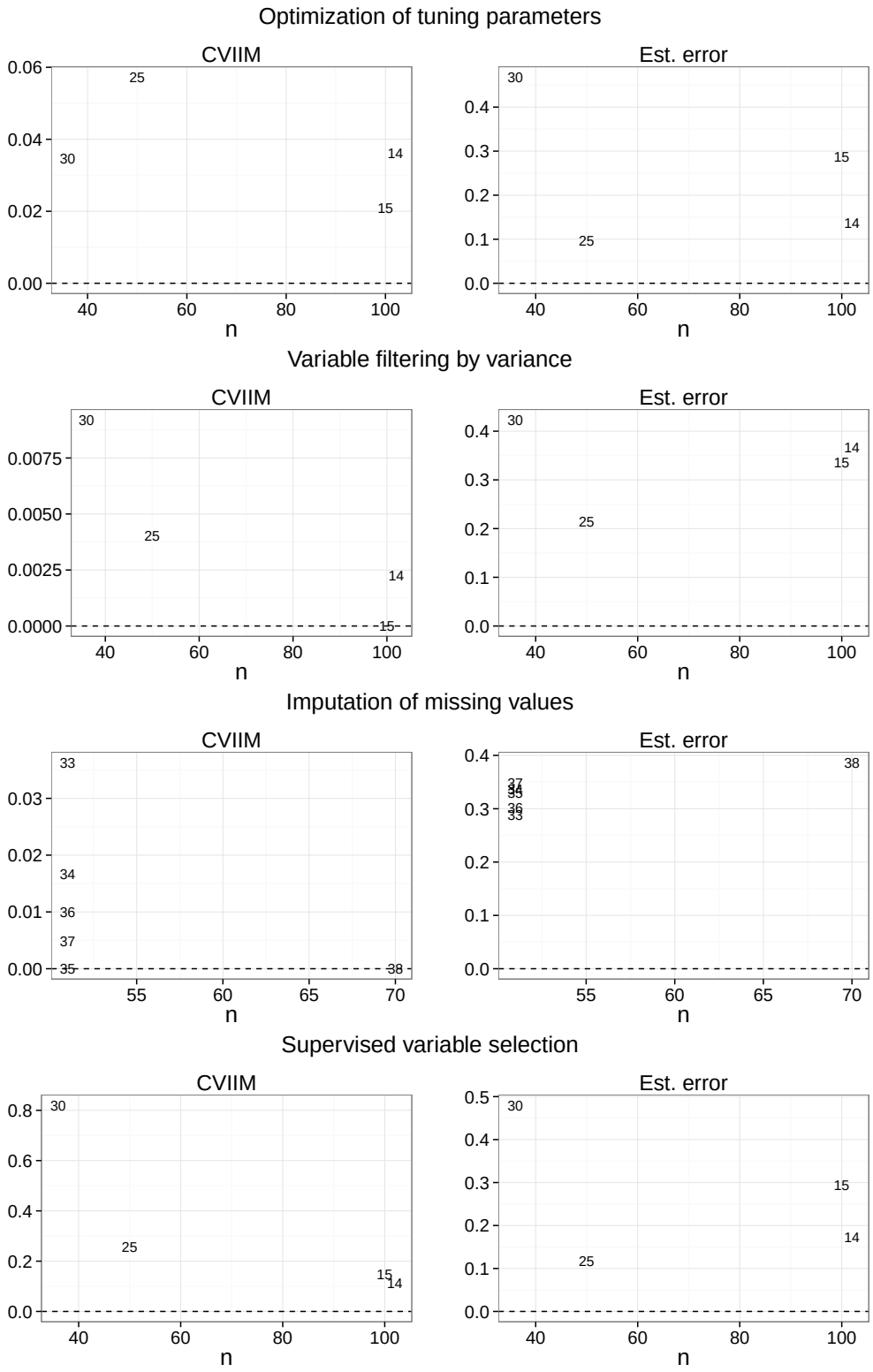

Figure 2.13.: Dataset-specific means of the $\text{CVIIM}_{\boldsymbol{s},n,K}$ and $e_{full,K}(\boldsymbol{s})$ values plotted against the sample sizes of the datasets

the misclassification errors in Eq. (2.1) with alternative error measures. The only requirement is that the loss function associated with the respective error type has a positive range. Most common loss functions fulfill this requirement, for example, the quadratic or absolute loss for linear regression, the integrated Brier score for survival data, the check function in the case of quantile regression, and the negative log-likelihood as an alternative to the error rate when the response variable is discrete.

Because CV provides dataset-internal error estimation, it estimates the error expected on observations following the same distribution as the training data. When a different dataset is used for evaluating the prediction rule as done in external validation, the error can be expected to be higher (Bernau et al.; 2014) (see also Chapter 4). CV can be used in the process of obtaining an adequate prediction rule when no external data are available; however, before ultimately applying a prediction rule in medical practice, it should be validated externally (Simon; 2004; Collins et al.; 2014).

2.5. Conclusions

In conclusion, the results of the empirical study in which the new measure of CV incompleteness is used suggest that 1) RMA and RMAglobalVSN can be performed safely as preliminary data preparation steps on the entire dataset since they yielded very small CVIIM values for all 12 real datasets analyzed, and 2) PCA has to be performed anew in each CV iteration, that is, re-trained on each training set, to avoid a potential optimistic bias, since it yielded large CVIIM values in some of the 20 real datasets analyzed. The latter result indicates that non-supervised data preparation steps also can lead to over-optimistic error estimation if performed before CV. Given the widespread use of RMA in microarray analysis, it is reassuring that the common practice of performing RMA before CV is not harmful.

Due to the complexity of modern biological data, traditional model assessment tools often are not appropriate or even employable and CV is the method of choice in evaluating prediction models. Thus, it is especially important to have reliable guidelines for its application. Moreover, data preparation is becoming increasingly important, especially for data generated by high-throughput technologies. The need to evaluate empirically the impact of CV incompleteness with regard to these data preparation steps likewise increases. In this chapter, through its application to important data preparation steps, CVIIM was shown to be a useful tool in this endeavor.

3. Combining location-and-scale batch effect adjustment with data cleaning by latent factor adjustment

3.1. Background

In practical data analysis, groups of observations included in a dataset sometimes form distinct batches; for example, measured at different times, under different conditions, by different people, or even in different laboratories. Such batch data are common in the context of high-throughput molecular data analysis, where experimental conditions typically have a high impact on the measurements and only few patients are considered at a time. Further, different batches may represent different studies concerned with the same biological question of interest. In this thesis all systematic differences between batches of data unattributable to the biological signal of interest are denoted as batch effects regardless of the context. If ignored when conducting analyses of the combined data, batch effects can lead to distorted and less precise results.

It is clear that batch effects are more severe when the sources from which the individual batches originate are more disparate. Batch effects, as explained in the definition above and as stated in Chapter 1, also may include systematic differences between batches due to biological differences among the respective populations unrelated to the biological signal of interest. This conception of batch effects is related to an assumption made about the distribution of the data on recruited patients in randomized controlled clinical trials (see, e.g., Matthews (2006)). This assumption is that the distribution of the (metric) outcome variable may be different for the actual recruited patients than for the patients eligible for the trial, that is, there may be biological differences, with one important restriction: The difference between the means in treatment and control group must be the same for recruited patients and

for eligible patients. Here, the population of recruited patients and the population of eligible patients can be perceived as two batches (ignoring that the former is a—very small—subset of the latter) and the difference between the means of the treatment and control group would correspond to the biological signal.

Throughout this chapter it is assumed that the data of interest are high-dimensional, that is, there are more variables than observations, and that all measurements are (quasi-)continuous. Possible present clinical variables are excluded from batch effect adjustment. Various methods have been developed to correct for batch effects (see e.g., Lazar et al. (2012) for a general overview and Luo et al. (2010) for an overview of methods suitable in applications involving prediction). Two of the most commonly used methods are ComBat (Johnson et al.; 2007), a location-and-scale batch effect adjustment method and SVA (Leek and Storey; 2007; Parker et al.; 2014), a non-parametric method in which the batch effects are assumed to be induced by latent factors. Even though the assumed form of batch effects underlying a location-and-scale adjustment as done by ComBat is rather simple, this method reduces batch effects (Chen et al.; 2011) considerably. However, a location-and-scale model often is too simplistic to account for more complicated batch effects. SVA is, unlike ComBat, concerned with situations where it is unknown which observations belong to which batches. The aim of this method is to remove inhomogeneities within the dataset that distort its correlation structure. These inhomogeneities are assumed to be caused by latent factors. When the batch variable is known, it is natural to take this important information into account when correcting for batch effects. Also, it is reasonable here to make use of the data-cleaning ability of the latent-factor adjustment by applying it within batches. This has the effect of reducing such inhomogeneities within batches which are unrelated to the biological signal of interest. By doing so it can be expected that the homogeneity of the data is further increased across batches as well.

In this chapter, a new method is introduced, denoted as *FAbatch*, where *FA* stands for *factor adjustment*. The method combines the location-and-scale adjustment (as performed by ComBat) with data cleaning by latent factor adjustment (as performed by SVA). Care has to be taken in the latent factor estimation in the context of data cleaning. Inhomogeneities within the dataset are induced not only by sources of unwanted noise but also by the biological signal of interest. If this interference between batch effects and signal would not be taken into account, removing the corresponding estimated latent factor loadings would lead to removing a large part of the biological signal of interest. An obvious, yet problematic, way of protecting the signal of interest would be to remove it temporarily before estimating the latent factors by regressing each of the variables in the dataset on the variable representing the biological signal. However, this can lead to an artificially increased signal, as outlined in

section 3.2.2. A solution in the case of a binary variable representing the biological signal is realized by FAbatch. Here, first, preliminary L_2-penalized logistic regression models are fitted to predict the probabilities of the individual observations to belong to the first class and to the second class. Second, these predicted probabilities are used in place of the actual values of the binary variable when protecting the signal of interest during latent factor estimation (see section 3.2.2 for details). Thus, in its current form FAbatch is applicable only when the signal variable is binary; however, extensions to other types of variables are possible (see section 3.4).

As an illustration, Figure 3.1 shows plots of the first two principal components obtained by PCA on a raw dataset (upper-left) and after running the three different batch effect adjustment methods described above. The dataset, composed of two batches, contains the gene expressions of 20 alcoholics and 19 healthy controls (downloadable from ArrayExpress (Kolesnikov et al.; 2015), accession number: E-GEOD-44456). After ComBat adjustment, the centers of gravity of the first principal components separated into the two batches become very similar (upper-right panel). However, the shapes of the point clouds corresponding to the two batches do not change substantially in comparison to the results obtained for the raw data (upper-left panel) and the two clouds do not fully overlap. After SVA adjustment, as with ComBat, the two batch centers are similar (lower-left panel), and the forms of the point clouds change considerably more than those with ComBat. Nevertheless, there still are regions in the plots with suboptimal overlap between the two clouds. The two batch centers are not distinguishable in the plot showing the result obtained after applying FAbatch (lower-right panel). Moreover, there is great overlap between the two clouds. This illustrative example suggests that the adjustment for batch effects can be improved by combining location-and-scale adjustment with data cleaning by factor adjustment.

As stated before, the prediction of phenotypes by means of prediction rules is an important area of application for high-throughput molecular data. In practice, the training data used to obtain the prediction rule often stem from a source different from that of the test data to which the prediction rule is applied. Batch-effect adjustment can be performed to render the test data more similar to the training data before applying a prediction rule that previously had been fitted on the training data. This will be empirically studied in depth in a realistic setting in Chapter 4. Note again that this kind of batch effect adjustment is not specific to FAbatch, but represents a general concept denoted as "addon batch effect adjustment": First, batch effect adjustment is conducted on the original dataset. Some methods require the values of the target variable to be known in the dataset under investigation. Second, batch-effect adjustment for independent batches is performed. To facilitate this, several

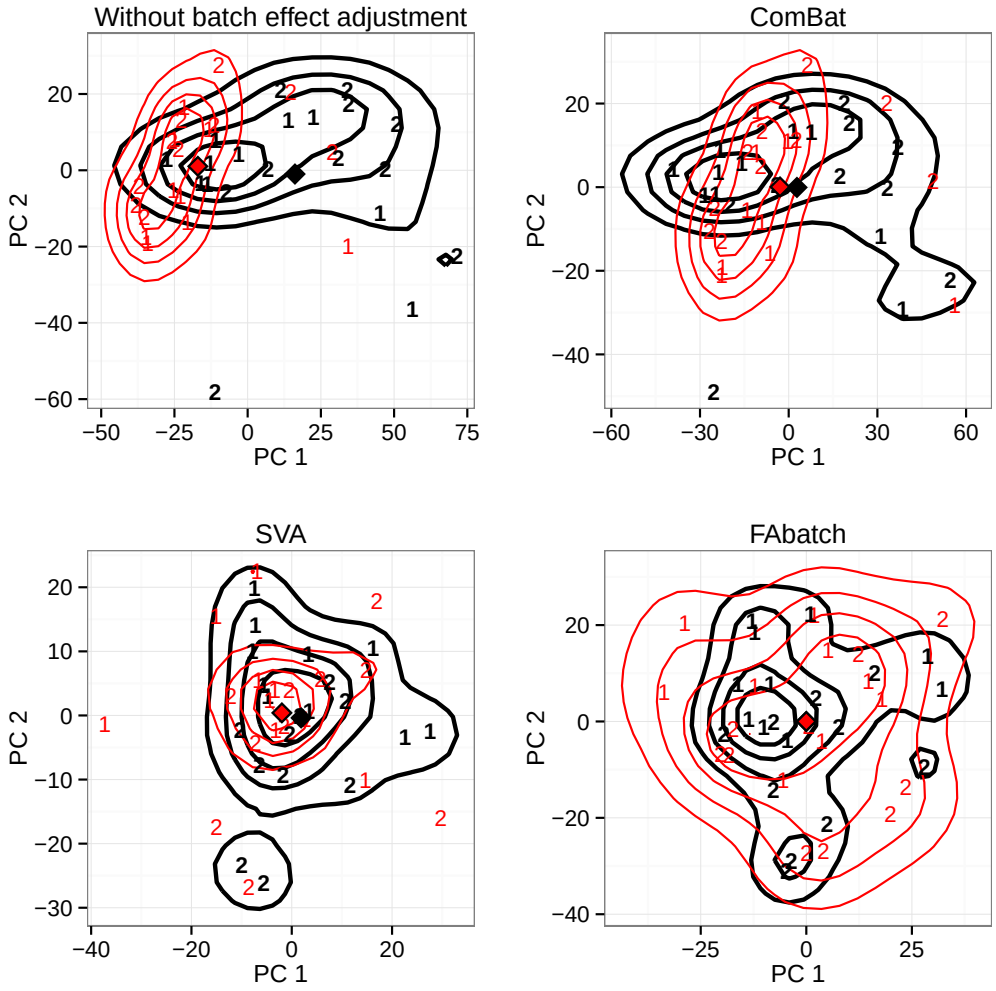

Figure 3.1.: Visualization of batch effect adjustment. First two principal components from PCA performed on the covariate matrix of a microarray dataset studying alcoholism: raw, after batch-effect adjustment according to ComBat, SVA using three factors, and FAbatch using three factors. The first batch is depicted in bold and the numbers distinguish the two classes "alcoholic" (2) versus "healthy control" (1) . The contour lines represent batch-wise two-dimensional kernel estimates and the diamonds represent the batch-wise centers of gravities of the points.

observations from each batch must be available simultaneously (in general, however not in the case of *frozen SVA* (fSVA), see section 3.2.3). This second phase does not affect the data prepared during the first phase (see section 3.2.3 for details). Such scenarios are referred to as *cross-batch prediction* in the rest of this chapter.

The structure of this chapter is as follows: Section 3.2 begins with an overview of the ComBat method, the SVA method, as well as other batch-effect adjustment methods. Subsequently, the new method is introduced and shown to be able to be seen as an extension of ComBat by batchwise adjustment for latent factor influences with this adjustment being similar to the application of SVA within batches. Moreover, how to adjust for batch effects a posteriori in independent observations for the purpose of cross-batch prediction is explained. The corresponding procedure is outlined in general and for the specific batch-effect adjustment methods considered in this chapter. In the subsequent subsection the design of an extensive comparison study based on simulations and real data applications is presented. In this study, FAbatch is compared with commonly used competitors with respect to diverse metrics measuring the effectiveness of batch-effect adjustment (Lazar et al.; 2012; Lee et al.; 2014). While the main aim here is to study the performance of FAbatch, the results of this comparison study also can be used to aid researchers in choosing appropriate batch-effect adjustment methods for their applications. The methods considered are FAbatch (`fabatch`), ComBat (`combat`), SVA (`sva`), mean-centering (`meanc`), standardization (`stand`), ratio-A (`ratioa`), and ratio-G (Luo et al.; 2010) (`ratiog`). The results of this study are described in section 3.3, which also contains an analysis of the use of batch-effect adjustment methods in cross-batch prediction. Moreover, it is argued that SVA can lead to an artificial increase of the biological signal of interest. The latter is demonstrated using simulated data. In section 3.4 the models behind FAbatch and other approaches are reviewed, and in section 3.5 important conclusions drawn from the results presented in this chapter are summarized.

3.2. Methods

3.2.1. Description of existing batch-effect adjustment methods

ComBat

This method assumes the following model for the data observed x_{ijg}:

$$x_{ijg} = \alpha_g + \boldsymbol{a}_{ij}^T \boldsymbol{\beta}_g + \gamma_{jg} + \delta_{jg}\epsilon_{ijg}, \quad \epsilon_{ijg} \sim N(0, \sigma_g^2). \tag{3.1}$$

Here i is the index for the observation, j the index for the batch, and g the index for the variable. The term $\boldsymbol{a}_{ij}^T\boldsymbol{\beta}_g$ parametrizes the effect of experimental conditions or, in general, any factors of interest $\boldsymbol{a}_{ij}$ on the measurements of variable g. In the context of this chapter, $\boldsymbol{a}_{ij}$ is a dummy variable representing the binary variable of interest y_{ij}, with $\boldsymbol{a}_{ij} = 1$ if $y_{ij} = 2$ and $\boldsymbol{a}_{ij} = 0$ if $y_{ij} = 1$. The term ϵ_{ijg} represents random noise unaffected by batch effects. The term γ_{jg} corresponds to the shift in location of variable g in the j-th batch; to the unobserved—hypothetical—data x^*_{ijg} unaffected by batch effects. The term δ_{jg} corresponds to the scale shift of the residuals for variable g in the j-th batch. Note that the restriction to binary target variables, which is not necessary in general for the application of ComBat, is required for the application of FAbatch (in its present form).

The unobserved counterpart x^*_{ijg} of x_{ijg} not affected by batch effects is assumed to be

$$x^*_{ijg} = \alpha_g + \boldsymbol{a}_{ij}^T\boldsymbol{\beta}_g + \epsilon_{ijg}, \quad \epsilon_{ijg} \sim N(0, \sigma_g^2). \tag{3.2}$$

The goal of batch effect correction via ComBat is to estimate these unobserved x^*_{ijg} values. The following transformation of the observed x_{ijg} values would provide the true x^*_{ijg} values:

$$\sqrt{\mathrm{Var}(x^*_{ijg})}\left(\frac{x_{ijg} - \mathbb{E}(x_{ijg})}{\sqrt{\mathrm{Var}(x_{ijg})}}\right) + \mathbb{E}(x^*_{ijg}) \tag{3.3}$$

$$= \sigma_g\left(\frac{x_{ijg} - (\alpha_g + \boldsymbol{a}_{ij}^T\boldsymbol{\beta}_g + \gamma_{jg})}{\delta_{jg}\sigma_g}\right) + \alpha_g + \boldsymbol{a}_{ij}^T\boldsymbol{\beta}_g \tag{3.4}$$

$$= \alpha_g + \boldsymbol{a}_{ij}^T\boldsymbol{\beta}_g + \epsilon_{ijg} = x^*_{ijg}. \tag{3.5}$$

In practice, however, the parameters involved in Eq. (3.4) are unknown and have to be estimated. In particular, γ_{jg}/σ_g and δ_{jg} are estimated using empirical Bayes to obtain more robust results (see Johnson et al. (2007) for details on the estimation procedure). Note that in the analyses performed and presented in this chapter the term $\boldsymbol{a}_{ij}^T\boldsymbol{\beta}_g$ is not included in the adjustment. The first reason for this is that in section 3.3.2 cross-batch prediction is used—the class values $\boldsymbol{a}_{ij}$ are not known in the test data when performing cross-batch prediction. The second reason is that using the class values $\boldsymbol{a}_{ij}$ with the estimates of $\boldsymbol{\beta}_g$ may lead to an artificially increased class signal because the estimates of $\boldsymbol{\beta}_g$ depend on the class values $\boldsymbol{a}_{ij}$. This kind of mechanism is discussed in detail, but in slightly other contexts, in sections 3.2.2 and 3.3.3.

SVA

The model for the observed data is given by:

$$x_{ijg} = \alpha_g + \boldsymbol{a}_{ij}^T \boldsymbol{\beta}_g + \sum_{l=1}^{m} b_{gl} Z_{ijl} + \epsilon_{ijg}, \tag{3.6}$$

$$\mathrm{Var}(\epsilon_{ijg}) = \sigma_g^2. \tag{3.7}$$

Here α_g and $\boldsymbol{a}_{ij}^T \boldsymbol{\beta}_g$ are as in the previous subsection and $Z_{ij1}, \ldots, Z_{ijm}$ are random latent factors with loadings $b_{g1}, \ldots, b_{gm}$.

The unobserved, corresponding, batch-free data are:

$$x_{ijg}^* = \alpha_g + \boldsymbol{a}_{ij}^T \boldsymbol{\beta}_g + \epsilon_{ijg}, \quad \mathrm{Var}(\epsilon_{ijg}) = \sigma_g^2. \tag{3.8}$$

Note again that in the SVA model batch membership is assumed to be unknown. To judge the appropriateness of the SVA algorithm it is important to specify the model underlying SVA as precisely as possible. From the following two facts it can be deduced that the distribution of the latent factors can be different for each observation—in the extreme case. First, the assumed form of the batch-free data in Eq. (3.8) implies that the distortions between the batches are induced fully by the latent factors. Second, each observation may come from a different batch with its own mean structure, covariance structure, and correlation structure.

The SVA batch-effect adjustment is performed by substracting $\sum_{l=1}^{m} b_{gl} Z_{ijl}$ from x_{ijg}:

$$x_{ijg} - \sum_{l=1}^{m} b_{gl} Z_{ijl} = \alpha_g + \boldsymbol{a}_{ij}^T \boldsymbol{\beta}_g + \epsilon_{ijg} = x_{ijg}^*. \tag{3.9}$$

The latent factors are estimated as the first m right singular vectors from a SVD. In section 3.1 it was stressed that inhomogeneities in datasets are due not only to batch effects, but also to the biological signal of interest, that is, the term $\boldsymbol{a}_{ij}^T \boldsymbol{\beta}_g$ in Eq. (3.6) and (3.8). Therefore, it was noted that the biological signal of interest has to be protected during factor estimation in FAbatch. In SVA, to protect the biological signal, before performing the SVD on the transposed covariate matrix, the variable values are weighted by the estimated probability that the corresponding variables are associated with unmeasured confounders but not with the binary variable representing the biological signal. The factor loadings are estimated by linear models. An extension of SVA (Leek and Storey; 2007) developed for the purpose of prediction is the fSVA procedure (Parker et al.; 2014), which will be explained in section 3.2.3.

Further batch effect adjustment methods considered in comparison studies

In the following, further batch-effect adjustment methods are presented which are less closely related to FAbatch than ComBat and SVA are, but they will be used in the analyses presented in this chapter and in Chapter 4.

Mean-centering From each measurement the mean of the values of the corresponding variable in the corresponding batch is substracted:

$$\widehat{x^*_{ijg}} = x_{ijg} - \widehat{\mu_{jg}}, \tag{3.10}$$

where $\widehat{\mu_{jg}} = (1/n_j) \sum_i x_{ijg}$.

Standardization The values of each variable are centered and scaled for each batch:

$$\widehat{x^*_{ijg}} = \frac{x_{ijg} - \widehat{\mu_{jg}}}{\sqrt{\widehat{\sigma^2_{jg}}}}, \tag{3.11}$$

where $\widehat{\mu_{jg}}$ as in (3.10) and $\widehat{\sigma^2_{jg}} = [1/(n_j - 1)] \sum_i (x_{ijg} - \widehat{\mu_{jg}})^2$.

Ratio-A Each measurement is divided by the arithmetic mean of the values of the variable in the corresponding batch:

$$\widehat{x^*_{ijg}} = \frac{x_{ijg}}{\widehat{\mu_{jg}}}, \tag{3.12}$$

where $\widehat{\mu_{jg}}$ is the same as in (3.10).

Ratio-G Each measurement is divided by the geometric mean of the values of the variable in the corresponding batch:

$$\widehat{x^*_{ijg}} = \frac{x_{ijg}}{\widehat{\mu_{g,geom}}}, \tag{3.13}$$

where $\widehat{\mu_{g,geom}} = \sqrt[n_j]{\prod_i^{n_j} x_{ijg}}$.

3.2.2. FAbatch

Model

The following model is assumed for the data observed x_{ijg}:

$$x_{ijg} = \alpha_g + \boldsymbol{a}_{ij}^T \boldsymbol{\beta}_g + \gamma_{jg} + \sum_{l=1}^{m_j} b_{jgl} Z_{ijl} + \delta_{jg} \epsilon_{ijg}, \tag{3.14}$$
$$Z_{ij1}, \ldots, Z_{ijm_j} \sim N(0,1), \quad \epsilon_{ijg} \sim N(0, \sigma_g^2),$$

Here, all parameters involved are the same as those in section 3.2.1. As in the SVA model, Z_{ijl} are random latent factors. In contrast to the SVA model, in the FAbatch model the distribution of the latent factors is the same for all observations. However, since the loadings b_{jgl} of the latent factors are batch-specific, they induce batch effects in the FAbatch model as well. More precisely, they lead to varying correlation structures in the batches. In the SVA model, by contrast, all batch effects are induced by the latent factors. Without the summand $\sum_{l=1}^{m_j} b_{jgl} Z_{ijl}$ model (3.14) would equal the model underlying the ComBat method (see Eq. (3.1)).

The unobserved data x_{ijg}^* unaffected by batch effects are assumed to have the following form:

$$x_{ijg}^* = \alpha_g + \boldsymbol{a}_{ij}^T \boldsymbol{\beta}_g + \epsilon_{ijg}, \quad \epsilon_{ijg} \sim N(0, \sigma_g^2). \tag{3.15}$$

Using estimated probabilities instead of actual classes

As noted in section 3.1, a further peculiarity of FAbatch is that the actual classes are not used when protecting the biological signal of interest in the estimation algorithm. Instead, probabilities of the observations to belong to either class are estimated and these are used in place of the actual classes.

The FAbatch procedure has two major advantages. First, it makes the batch-effect correction method applicable to prediction problems involving new test observations with unknown classes. Second, using the actual classes might lead to an artificial increase in separation between the two classes in the dataset. This is because, as will be seen in the next subsection, it is necessary to use the estimated class-specific means for centering the data before conducting factor estimation. Due to sampling variance, these estimated class-specific means often lie further away from each other than the true means, in particular for variables for which the true means lie close to each other. Subtracting the estimated influences of the factors leads to a reduction of the variance. If the variable values within the classes are centered before factor estimation is conducted, removing the estimated influences of factors would lead to a reduction in the variance around the respective estimated class-specific means. In

those—frequently occurring—cases, in which the estimated class-specific means lie further from each other than the corresponding true means, this would lead to an artificial increase in the discriminatory power of the corresponding variable in the adjusted dataset.

All analyses concerned with the discriminatory power of the covariate variables with respect to the target variable would be biased if performed on data adjusted in this way. More precisely, the discriminatory power would be overestimated. This mechanism is similar conceptually to the over-fitting of prediction rules to the data on which they were obtained. SVA suffers from a very similar kind of bias also related to using class information in protecting the biological signal (see section 3.3.3 for a detailed description of this phenomenon and the results of a small simulation study performed to assess the impact of this bias on data analysis in practice).

The probabilities of the individual observations to belong to the first and to the second class that are considered in FAbatch are estimated using models fitted from data other than the corresponding observations. Using these estimated probabilities instead of the actual classes attenuates the artificial increase in the class signal described above. The idea underlying the protection of the signal of interest is to center x_{ijg} before conducting factor estimation by subtracting the term

$$\begin{aligned}&\mathbb{E}(\alpha_g + \boldsymbol{a}_{ij}^T\boldsymbol{\beta}_g + \gamma_{jg}|x_{ij1,},\ldots,x_{ijp}) = \\ &\quad \Pr(y_{ij}=1|x_{ij1,},\ldots,x_{ijp})\ (\alpha_g+\gamma_{jg})+ \\ &\quad \Pr(y_{ij}=2|x_{ij1,},\ldots,x_{ijp})\ (\alpha_g+\boldsymbol{\beta}_g+\gamma_{jg}).\end{aligned} \tag{3.16}$$

Note that this adjustment is performed slightly differently in the FAbatch estimation algorithm.

Estimation

In the following, the estimation procedure of FAbatch is outlined:

1. Standardize the values x_{ijg} per batch:

$$x_{ijg,S} := \frac{x_{ijg} - \widehat{\mu_{jg}}}{\sqrt{\widehat{\sigma_{jg}^2}}}, \tag{3.17}$$

   where $\widehat{\mu_{jg}} = (1/n_j)\sum_i x_{ijg}$ and $\widehat{\sigma_{jg}^2} = [1/(n_j-1)]\sum_i(x_{ijg}-\widehat{\mu_{jg}})^2$. Here, the number of observations in batch j is denoted as n_j.

2. Using L_2-penalized logistic regression, estimate the probability of each obser-

vation to belong to the second class:

$$\widehat{\pi_{ij}} := \widehat{\Pr}(y_{ij} = 2|x_{ij1,S}, \ldots, x_{ijp,S}). \tag{3.18}$$

Here, the following CV-related procedure is employed. For batch $j \in \{1, \ldots, J\}$: 1) Fit a L_2-penalized logistic regression model using all observations except those in batch j; 2) Use the model fitted in step 1) to predict the probabilities π_{ij} of the observations from batch j. By using observations for fitting the models different from those used for predicting the probability overfitting is avoided in terms of the problems occurring when the actual classes are used as described in the previous subsection. The reason cross-batch prediction is performed here for estimating the probabilities instead of ordinary CV is that the resulting batch-adjusted data can be expected to be more suitable for the application in cross-batch prediction (see section 3.2.3). Here, to estimate the probabilities in the test batch a prediction model fitted on other batches has to be used. If the probabilities in the training data are estimated using ordinary CV, they will be more optimistic—that is, closer to zero and one, respectively—than those in the test data. This is because in ordinary CV observations from the same batch can be in training data and test data. By conducting cross-batch prediction for the estimation of the π_{ij} the situation encountered in cross-batch prediction applications is mimicked. The only, but important, exception where ordinary CV is performed to estimate the π_{ij} is when the data come from one batch only (this occurs in the context of cross-batch prediction when the training data consist of one batch, see also Chapter 4, where the latter occurs in the context of cross-study prediction).

The shrinkage intensity tuning parameter of the L_2-penalized logistic regression model is optimized with the aid of CV (Hsu et al.; 2010). For computational efficiency this optimization is not repeated in each iteration of the cross-batch prediction. Instead, it is performed beforehand on the complete dataset. Overoptimism resulting from this procedure compared to that resulting from "nested cross-batch prediction" is assumed to be negligible in the considered context.

3. Calculate the class adjusted values $x_{ijg,S,CA}$, which should contain considerably less class signal than $x_{ijg,S}$:

$$x_{ijg,S,CA} := x_{ijg,S} - (1 - \widehat{\pi_{ij}})\widehat{\mu_{g,S}}^{(1)} - \widehat{\pi_{ij}}\widehat{\mu_{g,S}}^{(2)}, \tag{3.19}$$

where $\widehat{\mu_{g,S}}^{(c)} = (1/\#L_c)\sum_{\{i^*,j^*\}\in L_c} x_{i^*j^*g,S}$ with $L_c = \{\{i,j\} : y_{ij} = c, i \in \{1,\ldots,n_j\}, j \in \{1,\ldots,J\}\}$ and $c \in \{1,2\}$.

4. Using $x_{ijg,S,CA}$, estimate the latent factors $Z^*_{ijm_j}$ and their loadings $b^*_{jgm_j}$ by the EM algorithm presented in Rubin and Thayer (1982), again considered by Friguet et al. (2009) in a specific context for microarray data. For the estimation of the number of factors see Friguet et al. (2009).

5. Subsequently the estimated factor contributions are removed:

$$x_{ijg,S,FA} := x_{ijg,S} - \widehat{b^*_{jg1}}\widehat{Z^*_{ij1}} - \cdots - \widehat{b^*_{jgm_j}}\widehat{Z^*_{ijm_j}}, \tag{3.20}$$

where $\widehat{b^*_{jg1}}, \ldots, \widehat{b^*_{jgm_j}}$ are the estimated, batch-specific factor loadings and $\widehat{Z^*_{ij1}}, \ldots, \widehat{Z^*_{ijm_j}}$ are the estimated latent factors. Note that only the factor contributions as a whole are identifiable, not the individual factors and their coefficients.

6. Finally, in each batch the $x_{ijg,S,FA}$ values are transformed to have the global means and pooled variances estimated before batch effect adjustment:

$$\widehat{x^*_{ijg}} = \left(\frac{x_{ijg,S,FA} - \widehat{\mu_{g,S,FA}}}{\sqrt{\widehat{\sigma^2_{g,S,FA}}}}\right)\sqrt{\widehat{\sigma^2_g}} + \widehat{\mu_g}, \tag{3.21}$$

where
$$\widehat{\mu_{g,S,FA}} = \Big(1/\sum_j n_j\Big)\sum_j\sum_i x_{ijg,S,FA},$$
$$\widehat{\sigma^2_{g,S,FA}} = \Big[1/\Big(\sum_j n_j - 1\Big)\Big] \sum_j\sum_i (x_{ijg,S,FA} - \widehat{\mu_{g,S,FA}})^2,$$
$$\widehat{\mu_g} = \Big(1/\sum_j n_j\Big)\sum_j\sum_i x_{ijg}$$
and
$$\widehat{\sigma^2_g} = \Big[1/\Big(\sum_j n_j - 1\Big)\Big]\sum_j\sum_i (x_{ijg} - \widehat{\mu_g})^2.$$

Note that by forcing the empirical variances in the batches to be equal to the pooled variances estimated before batch effect adjustment the residual variances σ^2_g in (3.14) are overestimated. This is because it is not taken into account that the variance is reduced by the adjustment for latent factors. However, unbiasedly estimating σ^2_g appears to be difficult due to scaling before estimation of the latent factor contributions.

Verification of model assumptions on real data

Due to the flexibility of its model FAbatch should—from a theoretical point of view—adapt well to real datasets. Nevertheless, it is important to test its validity on real data because the behavior of high-dimensional biomedical data does not become apparent through mere theoretical considerations. Therefore, in the following it is demonstrated that the model underlying FAbatch is indeed suited for real data using the dataset `BreastCancerConcatenation` presented in Table 3.1. This dataset was chosen because the batches involved are independent datasets in themselves and so the batch effects can be expected to be especially strong. When analyzing other datasets than `BreastCancerConcatenation` the same conclusions as for this dataset could be drawn (results not shown). Because the FAbatch model is an extension of the ComBat model by the addition of batch-specific latent factor contributions, the model fit of FAbatch is in the following compared to that of ComBat.

Figures B.1 and B.2 show a plot of the data values for each batch against the corresponding fitted values of FAbatch and ComBat. While there seem to be no deviations in the mean for either method, the association between data values and predictions is a bit stronger for FAbatch, except in the case of batch 4. This stronger association between fitted values and predictions for FAbatch can be explained by the fact that the factor contributions absorb part of the variance of the data values. In the case of batch 4, the estimated number of factors was zero, which explains why here the association is not stronger for FAbatch than for ComBat. Figures B.3 and B.4 correspond to the previous two figures, except that here the deviations from the fitted values instead of the data values are plotted against the corresponding fitted values. As illustrated, for batches 2, 3, and 5 the variance of these residuals depends slightly less on the mean for FAbatch than on the mean for ComBat. Batchwise density estimates of these residuals divided by their standard deviations are shown in Figures B.5 and B.6 for FAbatch and ComBat, respectively. For both methods outliers are observed. However, the distributions of the residuals differ between the two methods. In the case of ComBat the distributions are skewed for part of the batches, slightly for batches 3 and 5 and more strongly for batch 2. In the case of FAbatch the distributions are symmetric. A probable reason for the skewness of the distributions in the case of ComBat is that the residuals still contain the biological signal, as it is not included in the fixed part of the model.

3.2.3. Addon adjustment of independent batches

As described in section 3.1, an important feature of batch-effect adjustment methods is that they offer the possibility of making test data more similar to training

data of the same kind when studying the same biological issue of interest. Here, the training data and the test data both may consist of different batches. This feature of batch-effect adjustment can be used for prediction purposes in particular. In the following, first, an explanation is given of how batch effect adjustment is conceptionally performed to incorporate independent batches in general. Second, the respective procedures for the particular methods considered in this chapter are outlined.

General procedure

A batch-effect adjustment method (implicitly or explicitly) assumes a specific model for the data being observed. One part of the parameters involved in this model is connected with the data observed within the batches x_{ijg} and the other part with the unobserved batch-effect-free data x^*_{ijg}. While the values of the former kind of parameters in most cases depend on the individual batches, the values of the latter kind are the same for all observations, that is, they are batch-unspecific. When incorporating independent batches after having adjusted the training data, it is of interest to transform the data in the independent batches in such a way that their distribution becomes similar to those in the already adjusted training data without having to change the adjusted training data. This can be achieved by performing the same kind of transformation on the independent batches with the peculiarity that for the batch-unspecific parameters involved the estimates obtained for the training data are used. These procedures are referred to as addon batch effect adjustment procedures in the following.

Following the above definition, the batch effect adjustment methods which do not involve batch-unspecific parameters remain unchanged in addon batch effect adjustment. From the batch effect adjustment methods considered in this thesis, this is the case for mean-centering, standardization, ratio-A and ratio-G. In these methods the batch effect adjustment is performed batch by batch. The adjustment according to ComBat, FAbatch, and SVA does, by contrast, involve estimated batch-unspecific parameters.

ComBat

For ComBat, Luo et al. (2010) present the addon procedure for the situation of having only one batch in the training data. The addon batch effect adjustment with ComBat involves applying the standard ComBat adjustment to the validation data without the term $\boldsymbol{a}_{ij}^T\boldsymbol{\beta}_g$ and with all batch-unspecific parameters α_g, σ_g^2 and $\boldsymbol{\beta}_g$ estimated using the training data.

M-ComBat (Stein et al.; 2015) is a similar method applicable in the situation

of having one batch in the training data. This method can be used to perform a location-and-scale adjustment of the validation data, that is, in contrast to original ComBat this method does not involve shrinkage by empirical Bayes. Thus, according to the aforementioned definition of addon batch effect adjustment, M-ComBat represents the addon batch effect adjustment procedure for location-and-scale batch effect adjustment with one batch in the training data.

FAbatch

Adjustment with FAbatch involves estimates of the same batch-unspecific parameters as those with ComBat (according to Eq. (3.4)): α_g, σ_g^2 and $\boldsymbol{\beta}_g$. However, unlike in adjustment with ComBat, in FAbatch the term $\boldsymbol{a}_{ij}^T\boldsymbol{\beta}_g$ also is considered. This is achieved, basically, by estimating $\mathbb{E}(\boldsymbol{a}_{ij}|x_{ij1,},\ldots,x_{ijp})$ and $\boldsymbol{\beta}_g$ using L_2-penalized logistic regression (see section 3.2.2 for details). The addon procedure for FAbatch is derived straightforwardly from the general definition of addon procedures given above: The estimation scheme in section 3.2.2 is performed with the peculiarity that for all occurring batch-unspecific parameters, the estimates obtained in the adjustment of the training data are used.

SVA

There is a specific procedure for SVA denoted as fSVA (Parker et al.; 2014) for preparing independent data for prediction. More precisely, Parker et al. (2014) describe two versions of fSVA: the *exact fSVA algorithm* and the *fast fSVA algorithm*. The b_{gl}- and the $\boldsymbol{\beta}_g$ values are two of the batch-unspecific parameters involved in SVA adjustment. The $\boldsymbol{\beta}_g$ values are implicitly involved, namely when multiplying the variable values by the estimated probabilities that the corresponding variable is associated with unmeasured confounders, but not with the binary variable representing the biological signal. In both fSVA algorithms, when adjusting for batch effects in new observations the estimates of the b_{gl} values obtained for the training data are used. Also, for multiplying the variable values of a new observation by the estimated probabilities that the corresponding variable is associated with unmeasured confounders but not with the target variable, both algorithms use the estimates obtained for the training data. The distinguishing feature between the two algorithms is the way estimates of the factors Z_{ijl} for new observations are obtained.

In the first fSVA algorithm, denoted as exact fSVA algorithm in Parker et al. (2014), the latent factor vector for a new observation is estimated in the following way: 1) Combine the training data with the values of the new observation and multiply by the probabilities estimated on the training data; 2) Re-perform the SVD on the combined

data from 1) and use the right singular vector corresponding to the new observation as the estimate of its vector of latent factors. This algorithm is not an addon procedure. In this algorithm, the estimate of the latent factor vector for the test observation originates from a SVD different from that from which the estimated latent factors of the training observations originate. Therefore, this new estimated latent factor behaves, at least to some extent, differently than those of the training data. As a consequence, when adjusting the new observation, a feature of addon procedures is not given: The same kind of transformation must be performed for independent batches. This problem can be assumed to have a lower impact for larger training datasets. Here, the latent factor model estimated on the training data depends less on whether or not a single new observation is included in the SVD. A solution to the problem of differently behaving latent factor estimates in training data and test data would be the following: To adjust the training data use the estimates of the latent factors (and their loadings) obtained in the second SVD performed after including the test observation. However, this again would not correspond to an addon procedure because the adjusted training data would change each time a new observation is included, which is not allowed, as stated in the definition of addon procedures given in section 3.2.3.

The second fSVA algorithm, denoted as fast fSVA algorithm in Parker et al. (2014) takes a different approach. Here, the SVD is not re-performed entirely on the combination of the training data and the new observation. Instead, basically a SVD is performed in order to calculate the right singular vector corresponding to the new observation under the restriction that the left singular vectors and singular values are fixed to the values of the parameters obtained from the SVA performed on the training data. Thus in this adjustment, it is taken into account that the left singular vectors and singular values are batch-unspecific parameters. The resulting estimated latent factor vector of the new observation behaves in the same way as that of the training data, because here it originates from the same SVD. This algorithm does correspond to an addon procedure because the same kind of transformation is performed for independent batches, or rather independent observations in the SVA model, without the need to change the training data.

The fSVA algorithms initially seem intuitive. However, when using the estimated factor loadings (and other information in the case of the fast fSVA algorithm) from the training data the same sources of heterogeneity must be present in the training data and test data, which might not be true in case of a test data batch from a different source. Thus, fSVA is fully applicable only when training data and test data are similar, as stated by Parker et al. (2014). Nevertheless, in section 3.3.2 it is applied in cross-batch prediction to obtain indications of whether the prediction

performance of classifiers might even deteriorate through the use of fSVA when the training data and test data are very different.

Above, I have presented the addon procedures for the batch effect adjustment methods considered in this thesis. However, adhering to the general definition of addon procedures, such algorithms can be derived readily for other methods as well.

3.2.4. Comparison of FAbatch to other methods

A comprehensive evaluation of the ability of FAbatch to adjust for batch effects in comparison to its competitors was performed, using both simulated datasets and real datasets. Simulation makes it possible to study the performance subject to basic settings and to use a large number of datasets. Nevertheless, simulated data never can capture all properties found in real datasets from the area of the application. Therefore, in addition, 14 publicly available real datasets were studied, each consisting of at least two batches.

The value of batch effect adjustment contains various aspects connected with the adjusted data or with the quality of results of analyses performed using the adjusted data. Therefore, when comparing batch effect adjustment methods, it is necessary to consider several criteria, each of which must be concerned with a certain aspect. Seven metrics were calculated to measure the performance of each batch effect adjustment method on each simulated dataset and each real dataset.

In the following, first, the seven metrics considered in the comparison study described above are outlined. Second, the simulation designs are explained and basic information on the real datasets is provided. Third, the results of these analyses are interpreted and presented in section 3.3.1.

Performance metrics

In this section the performance metrics used to assess batch effect adjustment are described. Several of them are, in their original form, restricted to the case of two batches only. For datasets with more than two batches they are extended as follows: 1) Calculate the original metric for all possible pairs of batches and 2) calculate the weighted average of the values in 1) with weights proportional to the sum of the sample sizes in the two respective batches.

Separation score (`sepscore`) This metric was derived from the mixture score presented in Lazar et al. (2012), which was inapplicable here because it depends on the relative sizes of the two involved batches j and j^*. Generally, the mixture score measures the degree of mixing between the observations belonging to the two batches

after batch effect adjustment. By contrast, the separation score measures the degree of separation between the two batches. First, for each observation in j, its k nearest neighbors are determined in both batches simultaneously with respect to the euclidean distance. Here, the proportion of nearest neighbors belonging to batch j^* is calculated. Then, the average, denoted as MS_j, is taken of the n_j proportions obtained in this way. This value is the mixture score as presented in Lazar et al. (2012). To obtain a measure for the separation of the two batches the absolute difference between MS_j and its value expected in the absence of batch effects is taken: $|\,\mathrm{MS}_j - n_{j^*}/(n_j + n_{j^*} - 1)|$. The separation score is defined as the simple average of this absolute difference and the corresponding quantity when the roles of j and j^* are switched. The number of k nearest neighbors considered was set to 10. Smaller values of the separation score are better.

Average minimal distance to other batch `(avedist)` A very similar metric for two batches is the average minimal distance to the other batch after batch effect adjustment (see also Lazar et al. (2012)). For each observation in batch j the euclidean distance to the nearest observation in batch j^* is calculated. Consecutively, the roles of j and j^* are switched and finally the average is computed over all $n_j + n_{j^*}$ minimal distances. To obtain a metric independent of the scale, the variables are standardized before the calculation to have a zero mean and uniform variance. Here, smaller values are better.

Kullback-Leibler divergence between density of within- and between-batch pairwise distances `(klmetr)` This metric, used in Lee et al. (2014) in a similar form is, again, based on the distances of the observations within and between batches. First, the distances between all pairs of observations in batch j, denoted as $\{dist_j\}$, and the distances between all such pairs in batch j^*, denoted as $\{dist_{j^*}\}$, are calculated. Then, for each observation in j the distances to all observations in j^* are calculated, resulting in $n_j \times n_{j^*}$ distances denoted as $\{dist_{jj^*}\}$. Consecutively, the Kullback-Leibler divergence is estimated between the densities of $\{dist_j\}$ and $\{dist_{jj^*}\}$ and between the densities of $\{dist_{j^*}\}$ and $\{dist_{jj^*}\}$, using the k-Nearest Neighbors-based method by Boltz et al. (2009) with $k = 5$. Finally, the weighted mean of the values of these two divergences is calculated, with weights proportional to n_j and n_{j^*}. As in the case of `avedist` the variables are standardized before the calculation to make the metric independent of scale. Smaller values of this metric are better.

Skewness divergence score `(skewdiv)` This metric presented in Shabalin et al. (2008) is concerned with the values of the skewness of the observation-wise empirical

distributions of the data. Because batch effect adjustment should make the distribution of the data similar for all batches, these skewness values should not differ substantially across batches after a successful batch effect adjustment. The metric is obtained as follows for two batches j and j^* after batch effect adjustment: 1) For each observation in batch j and for each observation in batch j^* calculate the difference between the mean and the median of the data as a measure for the skewness of the distribution of the data values; 2) Determine the area between the two batch-wise empirical cumulative density functions of the values obtained from 1). The value obtained in 2) can be regarded as a measure for the disparity of the batches with respect to the skewness of the observation-wise empirical distributions. Again, standardization is conducted before the calculation. Smaller values indicate a more successful batch effect adjustment with respect to the homogeneity of the skewness values.

Proportion of variation induced by class signal estimated by principal variance component analysis `(pvca)` Principal variance component analysis (Li et al.; 2009) allows the estimation of the contributions of several sources of variability. Here, first PCA is performed on the $n \times n$ covariance matrix between the observations. Then, using a random effects model, the principal components are regressed on arbitrary factors of variability, such as "batch" and "(phenotype) class". Ultimately, estimated proportions of variance induced by each factor, and that of the residual variance are obtained; for details see Li et al. (2009). The following factors were included into the model: "batch", "class" and the interaction of "batch" and "class". As a metric the proportion of variance explained by "class" was used. Naturally, higher values of this metric indicate a better preservation or exposure, respectively, of the biological signal of interest.

Performance of differential expression analysis `(diffexpr)` This metric is similar to the idea presented in Lazar et al. (2012) which involves comparing the list of genes deemed differentially expressed the strongest when using a batch effect adjusted dataset to the corresponding list obtained using an independent dataset. Having no independent data available here, a slightly different approach was taken: 1) Omit each batch j and perform batch effect adjustment on the remaining batches. Derive two lists of the 5% of variables deemed differentially expressed the strongest (see next paragraph for details): one using the batch effect adjusted dataset, where batch j was omitted, and one using the data from batch j. Calculate the number of variables appearing in both lists and divide this number by the length of the lists. 2) Calculate a weighted average of the values obtained in 1) with weights proportional to the number of observations in the corresponding omitted batches. Note that in the case

of the simulated datasets it would have been possible to estimate the true discovery rate instead of calculating the metric described above. However, for the sake of comparability, the procedure described above was followed for the simulated data as well.

In the following the procedure performed to estimate the 5% of variables most differentially expressed is described. The original idea to use the p-values of simple two-sample t-tests between the two classes was soon discarded. The reason for this was that this procedure might have favoured batch effect adjustment methods that produce more normally distributed values of the variables. The p-values of classical non-parametric tests, such as the Mann-Whitney-Wilcoxon rank sum test also would have been unsuitable here because here the p-values can adopt a limited number of possible values only. Therefore, it would have occurred in many cases that more than 5% of the variables adopt the smallest of possible p-values, making a selection of 5% of variables with the smallest p-values impossible. As a solution, for each variable a randomized p-value from the Whitney-Wilcoxon rank sum test was drawn (see Geyer and Meeden (2005) for details). These randomized p-values can adopt any possible value between zero and one and therefore were suitable for ordering the variables according to their degree of differential expression between the two classes. Ultimately, the 5% of variables that were associated with the smallest p-values were considered. Higher values of this metric are better.

Mean Pearson's correlation of the variable values before and after batch effect adjustment `(corbeaf)` This metric suggested by Lazar et al. (2012) is not a measure for the performance of batch effect adjustment. However, it may be used occasionally to decide between two methods performing similarly: In such cases the method that least affects the data (i.e., that with smaller `corbeaf` values) could be preferred (Lazar et al.; 2012).

Simulation design

Three basic scenarios were considered: 1) Common correlation structure in all batches (ComCor); 2) Batch-specific correlation structures (BatchCor); 3) Batch- and class-specific correlation structures (BatchClassCor). For each of these three scenarios the correlations were induced in two ways (see below for details): 1) simulating from a latent factor model with normally distributed residuals and 2) drawing from multivariate normal distributions with specified correlation matrices. The second scheme was considered to avoid favoring FAbatch and SVA by restricting the simulation to factor-based data generation mechanisms. Each simulated dataset consisted of four

batches with 25 observations each. The number of variables was 1000. For each of the six (3×2) settings 500 datasets were simulated. The values of the parameters occurring in the simulation models were based on corresponding estimates obtained from two publicly available microarray datasets: a dataset also used in the real data study, denoted as `AutismTranscr` (Table 3.1), and a dataset on colon cancer, denoted as `ColoncbTranscr`. The latter is downloadable from ArrayExpress (Kolesnikov et al.; 2015), accession number: E-GEOD-44861.

All six settings can be expressed using the following general model:

$$\begin{aligned} \boldsymbol{x}_{ij} &= \boldsymbol{\alpha} + \boldsymbol{a}_{ij}\boldsymbol{\beta} + \boldsymbol{\gamma}_j + \boldsymbol{\epsilon}^*_{ij}, \\ \boldsymbol{\epsilon}^*_{ij} &\sim MVN(\mathbf{0}, \boldsymbol{\Sigma}_{j,\boldsymbol{a}_{ij}}), \end{aligned} \tag{3.22}$$

with $\boldsymbol{x}_{ij} = (x_{ij1}, \ldots, x_{ijp})^T$, $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_p)^T$, $\boldsymbol{a}_{ij} \in \{0, 1\}$, $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)^T$, $\boldsymbol{\gamma}_j = (\gamma_{j1}, \ldots, \gamma_{jp})^T$, $\boldsymbol{\epsilon}^*_{ij} = (\epsilon^*_{ij1}, \ldots, \epsilon^*_{ijp})^T$, $j \in \{1, \ldots, K\}$ and $p = 1000$.

The elements of $\boldsymbol{\alpha}$ and $\boldsymbol{\gamma}_j$ ($j \in \{1, \ldots, K\}$) were drawn from normal distributions with means and variances based on corresponding estimates obtained from `ColoncTranscr` (for details see the corresponding commented R code at the following link: `http://www.ibe.med.uni-muenchen.de/organisation/mitarbeiter/070_drittmittel/hornung/fabatchpaper_suppfiles/index.html`). The vector of the class differences $\boldsymbol{\beta}$ contained 300 (30%) non-zero values. Half of these were negative and half positive. The values were drawn from gamma distributions, where the choice of parameters was, again, based on `ColoncTranscr`. Here, in the case of the negative entries of $\boldsymbol{\beta}$, the sign of the originally drawn values was changed.

The six settings differed with respect to the specification of $\boldsymbol{\Sigma}_{j,\boldsymbol{a}_{ij}}$. The differences are outlined in the following.

Design A: Simulating from latent factor model The residuals of the fixed part of the model $\boldsymbol{\epsilon}^*_{ij}$ were simulated in the following ways for the corresponding scenarios:

1. ComCor:
$$\epsilon^*_{ijg} := \sum_{m=1}^{5} b_{0gm} Z_{ijm} + \delta_{jg}\epsilon_{ijg} \tag{3.23}$$

2. BatchCor:
$$\epsilon^*_{ijg} := \sum_{m=1}^{5} b_{0gm} Z_{ijm} + \sum_{m=1}^{5} b_{jgm} \overset{*}{Z}_{ijm} + \delta_{jg}\epsilon_{ijg} \tag{3.24}$$

3. BatchClassCor:
$$\epsilon^*_{ijg} := \sum_{m=1}^{5} b_{0gm} Z_{ijm} + \sum_{m=1}^{5} \widetilde{b}_{\boldsymbol{a}_{ij}gm} Z_{ijm} + \sum_{m=1}^{5} b_{jgm} \overset{*}{Z}_{ijm} + \delta_{jg}\epsilon_{ijg}, \tag{3.25}$$

where $\epsilon_{ijg} \overset{iid}{\sim} N(0, \sigma^2_g)$ and Z_{ijm}, $\overset{*}{Z}_{ijm} \overset{iid}{\sim} N(0,1)$. b_{0gm}, b_{jgm} and $\widetilde{b}_{\boldsymbol{a}_{ij}gm}$ were drawn from normal distributions and δ^2_{jg} and σ^2_g from inverse gamma distributions. Again, the parameters of the latter distributions were based on corresponding estimates obtained from `ColoncTranscr`.

In Eq. (3.23), (3.24), and (3.25) factors $Z_{ij1}, \ldots, Z_{ij5}$ model the biological correlation between the variables. Factors $\overset{*}{Z}_{ij1}, \ldots, \overset{*}{Z}_{ij5}$ in Eq. (3.24) and (3.25) model distortions that affect the correlation in the batches. In the ComCor setting all observations have the same correlation structure—independent of the batch. In the BatchCor setting the correlation structure is different in each batch due to the batch-specific loadings of factors $\overset{*}{Z}_{ij1}, \ldots, \overset{*}{Z}_{ij5}$. In the third setting, BatchClassCor, the correlations differ not only by batch but also according to which of the two classes the observations are in, that is, there are batch- and class-specific correlations. In each setting the variances are different in the batches.

Design B: Drawing from multivariate distributions with specified correlation matrices In Design B, all correlation matrices appearing in the three scenarios were estimated using real data. First, using the R function `cor()` an approximate positive definite correlation matrix was estimated and then the R function `nearPD()` from the R package `Matrix` was applied to the result to calculate the nearest positive definite correlation matrix. The 1000 genes from the `AutismTranscr` dataset showing

themselves to be the most related to the binary outcome according to variable-wise two-sample t-tests were used. Before estimating the correlation matrices, the data were further centered by class in each batch to adjust for excess correlations due to class differences. The variances are the same in all three scenarios. They were set to be equal to those in the ComCor setting of Design A, that is, $\sum_{m=1}^{5} b_{0gm}^2 + \delta_{jg}^2 \sigma_g^2$.

The correlation matrices were obtained as follows for the three settings:

1. ComCor: A single correlation matrix was used for all batches and was estimated from the data of a single batch in `AutismTranscr`.

2. BatchCor: A separate correlation matrix was used for each batch, each estimated from the data of a batch in `AutismTranscr`.

3. BatchClassCor: A separate correlation matrix was used for each combination of batch and class, where each was estimated on a corresponding batch-class combination in `AutismTranscr`.

After obtaining the correlation matrices, the corresponding covariance matrices were calculated by multiplying each entry in the correlation matrices with the respective pair of standard deviations.

Datasets

Fourteen high-dimensional datasets with a binary target variable and with at least two batches were downloaded from the ArrayExpress database (Kolesnikov et al.; 2015) and the NCBI GEO database (Barrett et al.; 2013). In searching for suitable datasets on ArrayExpress and NCBI GEO, the search term "batch" was entered and the search hits were surveyed manually. This procedure was chosen in order to maximize the number of possibly eligible datasets. Exclusion criteria were as follows: not enough samples, no batch variable, and no possibility to form a suitable binary target variable. The selection of the datasets was not in any way based on the results they yielded with the different methods, thus following Rule 4 from Boulesteix (2015): "do not fish for datasets".

Three datasets featured too many variables to be manageable from a computational point of view. Therefore, in these cases, 50,000 variables were randomly selected. When missing values occurred in a dataset the following approach was taken. First, variables with too many missing values were excluded. Second, the remaining missing values were imputed by the median of the observed values of the corresponding variable in the corresponding batch. This simplistic imputation procedure can be justified by the very low numbers of variables with missing values in all datasets.

Outlier analysis was performed by visually inspecting the principal components of the PCA applied to the individual datasets and suspicious samples were removed. Figure 3.2 shows the first two principal components of PCA applied to each of the datasets used after imputation and outlier removal.

Table 3.1 gives an overview on the datasets. Information on the nature of the binary target variable for each dataset is given in Appendix B.2. The dataset `BreastCancerConcatenation` is a concatenation of five independent breast cancer datasets. For the remaining 13 datasets the reason for the batch structure could be ascertained in four cases only. In three of these, batches were the result of hybridization and in one case labeling (for details see Appendix B.3).

Details regarding the background of the datasets can be found online on the ArrayExpress webpage using the corresponding accession numbers. Moreover, corresponding R scripts written for preparation of the datasets can be obtained from the following link: `http://www.ibe.med.uni-muenchen.de/organisation/mitarbeiter/070_drittmittel/hornung/fabatchpaper_suppfiles/index.html`. Here, the R code necessary to reproduce all analyses performed in this chapter also is provided.

3.3. Results

3.3.1. Ability to adjust for batch effects

Figures B.7 to B.13 show the values of the individual metrics obtained for the simulated data and Figure 3.3 shows the corresponding results obtained for the 14 real datasets. Tables B.1 to B.7 for the simulated data and Tables 3.2 and 3.3 for the real data show the means of the metric values separated by method (and simulation scenario) together with the mean ranks of the methods with respect to the individual metrics. In most cases, results of the study of the simulated data differ only slightly between the settings with respect to the ranking of the methods by their performance. Therefore, only occasionally will the scenarios in the interpretations be differentiated. In addition, analyses of the simulated data and real data often yield similar results. Differences will be discussed whenever relevant.

According to the values of the separation score (Figures B.7 and 3.3, Tables B.1 and 3.2) ComBat, FAbatch, and standardization seem to lead to the best mixing of the observations across the batches. For the real datasets, however, standardization was only slightly better on average than other methods.

The results with respect to `avedist` are less clear. For Design A the results of the simulation indicate that FAbatch and SVA are associated with greater minimal distances to neighboring batches than are the other methods. However, for Design

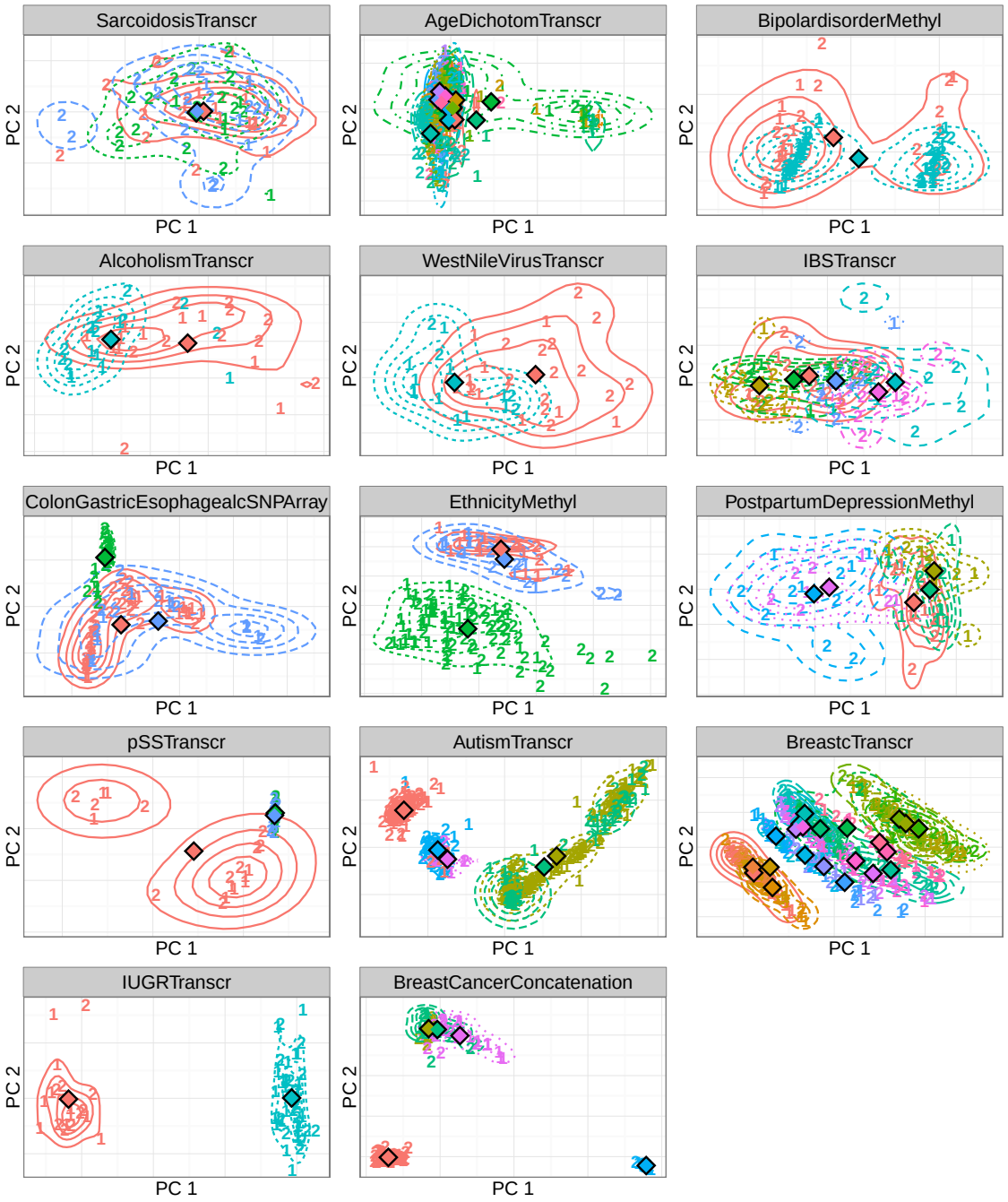

Figure 3.2.: Each subplot shows the first two principal components out of PCA performed on the covariate matrix of one of the datasets used. In each case the colors distinguish the batches, and the numbers distinguish the two classes "diseased" (2) versus "healthy control" (1). The contour lines represent batch-wise two-dimensional kernel estimates and the diamonds represent the batch-wise centers of gravities of the points. The plots are arranged in ascending order according to the strength of batch effects with respect to the following criterion: average over the euclidean distances between all possible pairs of points in the plot from different batches divided by the analoguous mean over all such pairs from the same batches.

Label	Num. of observ.	Num. of batches	Num. of variables	Prop. with disease	Data type	Source (Acc.num.)
ColonGastricEsophagealcSNPArray	93	3	50000	0.54	comparative genomic hybridization	ArrayExpr.: E-GEOD-36458
AgeDichotomTranscr	243	15	27568	0.49	DNA methylation profiling	ArrayExpr.: E-GEOD-36194
EthnicityMethyl	133	3	50000	0.45	DNA methylation profiling	ArrayExpr.: E-GEOD-39672
BipolardisorderMethyl	94	2	27537	0.50	DNA methylation profiling	ArrayExpr.: E-GEOD-38873
PostpartumDepressionMethyl	50	5	50000	0.46	DNA methylation profiling	ArrayExpr.: E-GEOD-44132
AutismTranscr	439	5	24526	0.53	transcription profiling	ArrayExpr.: E-GEOD-37772
BreastcTranscr	410	23	20180	0.50	transcription profiling	ArrayExpr.: E-GEOD-44281
BreastCancerConcatenation	168	5	22277	0.65	transcription profiling	ArrayExpr.: E-GEOD-27562, E-GEOD-21422, E-GEOD-22544, E-GEOD-20266, E-TABM-276
IUGRTranscr	67	2	48701	0.40	transcription profiling	ArrayExpr.: E-GEOD-35574
IBSTranscr	63	6	54671	0.70	transcription profiling	ArrayExpr.: E-GEOD-36701
SarcoidosisTranscr	58	3	54675	0.66	transcription profiling	NCBI GEO: GSE19314
pSSTranscr	49	3	54675	0.63	transcription profiling	ArrayExpr.: E-GEOD-40611
AlcoholismTranscr	39	2	28869	0.51	transcription profiling	ArrayExpr.: E-GEOD-44456
WestNileVirusTranscr	39	2	47323	0.46	transcription profiling	ArrayExpr.: E-GEOD-43190

Table 3.1.: Overview of datasets used in empirical studies. The following information is given: number of observations, number of batches, number of variables, proportion of observations with disease ($y = 2$), biomolecular data type, accession number.

B this is not clearly observed other than for the setting with common correlations. Results for the real data also indicate no clear ordering of the methods with respect to this metric (see in particular the means over the datasets in Table 3.2). The values of this metric were not appreciably improved by batch effect adjustment in general on the real datasets.

The values of `klmetric`, which conceptionally is very similar to the separation score, allow a very similar conclusion to be drawn as that from the latter metric (Figures B.9 and 3.3, Tables B.3 and 3.2): ComBat, FAbatch, and standardization performed the best. While this conclusion could be drawn about both simulated data and real data, other results of the simulation scenarios and the real data analyses differed: SVA performed considerably worse for Design A than for Design B and mean-centering performed better on the simulated data in general.

The estimates of the proportions of the variation explained by the class signals obtained through principal variance component analysis (`pvca`) are depicted in Figures B.10 and 3.3 and summarized in Tables B.4 and 3.2. SVA appears to be associated with the highest proportion of variation induced by the class signal; however, comparison to the other methods is unfair here: SVA makes use of the target variable and therefore is associated with an artificially increased class signal (see section 3.3.3 for details on this mechanism related to overoptimism). FAbatch performed well on the simulated data but not on the real datasets, where it had the lowest mean value with the exception of no batch effect adjustment. Figure 3.3 reveals that those three datasets for which `pvca` was considerably smaller after batch effect adjustment using FAbatch were, at the same time, the three datasets with the highest `pvca` values before batch effect adjustment. Datasets with high `pvca` values are datasets where the biological signal is relatively strong in comparison to the batch effects. The results suggest that for such datasets, batch effect adjustment with FAbatch might be counterproductive. The distinguishing feature of FAbatch as opposed to a mere location-and-scale adjustment as performed by ComBat is that it aims to adjust additionally for batch effects not explainable by location or scale shifts. While FAbatch aims to protect the biological signal in the factor estimation, this signal cannot be protected entirely here due to the uncertainty in the estimation of the class probabilities. When reducing the total heterogeneity by FAbatch in cases of weak batch effects, the merit of removing heterogeneity due to batch effects becomes smaller in comparison to the harm that affects the signal. ComBat performed better than other methods here on the real data (with the exception of SVA as mentioned before).

For the performance metric related to differential expression analysis `diffexpr` (Figures B.11 and 3.3, Tables B.5 and 3.3) the results for FAbatch and SVA differed substantially between simulated data and real data. In the simulation, these

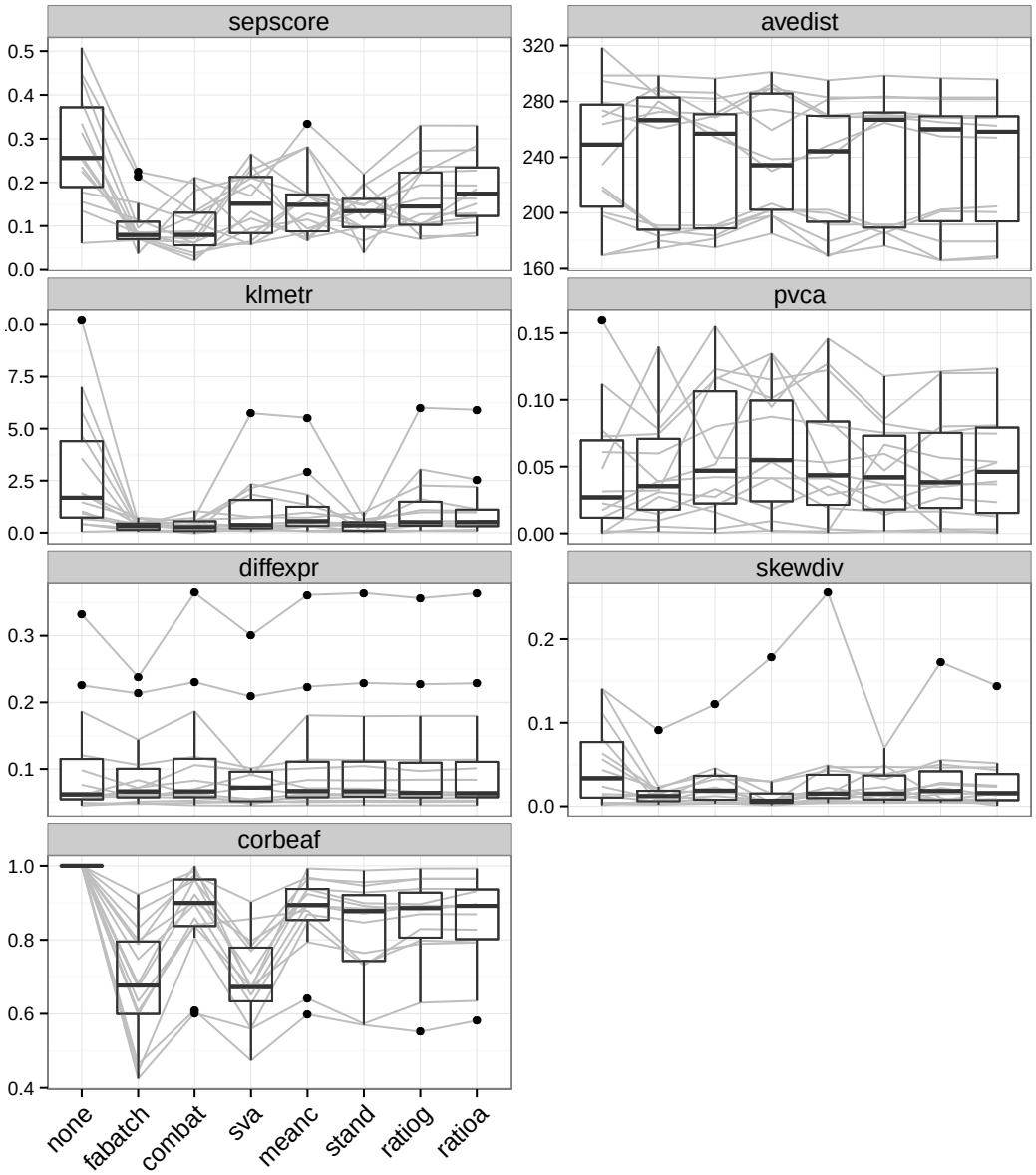

Figure 3.3.: Metric values in real datasets. Boxplots of values for all 14 datasets separated by method for the following metrics: `sepscore`, `avedist`, `klmetr`, `pvca`, `diffexpr`, `skewdiv`, and `corbeaf`. The grey lines connect values corresponding to the same datasets.

sepscore

mean values	combat 0.09895	**fabatch** **0.10227**	stand 0.13238	sva 0.15217	meanc 0.15807	ratiog 0.16618	ratioa 0.18314	none 0.2806
mean ranks	combat 2.28571	**fabatch** **3.35714**	stand 3.71429	sva 4.42857	meanc 4.64286	ratiog 4.78571	ratioa 5.92857	none 6.85714

avedist

mean values	meanc 233.32619	ratiog 235.27321	ratioa 235.39525	combat 235.52757	stand 237.86855	**fabatch** **239.53197**	sva 240.55365	none 243.10948
mean ranks	meanc 3.07143	combat 3.57143	ratiog 3.57143	ratioa 3.57143	**fabatch** **5.14286**	stand 5.21429	none 5.78571	sva 6.07143

klmetr

mean values	**fabatch** **0.32312**	combat 0.33748	stand 0.35524	sva 1.08835	meanc 1.13029	ratioa 1.15025	ratiog 1.23577	none 2.85956
mean ranks	combat 2.85714	**fabatch** **3**	stand 3.14286	sva 4.57143	meanc 4.71429	ratioa 4.92857	ratiog 5.42857	none 7.35714

pvca

mean values	sva 0.06364	combat 0.06015	meanc 0.05636	ratioa 0.0502	ratiog 0.04933	stand 0.04741	**fabatch** **0.04569**	none 0.04477
mean ranks	sva 2.92857	combat 3.14286	meanc 3.57143	ratioa 5	stand 5.07143	ratiog 5.21429	**fabatch** **5.35714**	none 5.71429

Table 3.2.: Means of the metric values and of the corresponding ranks among the various methods for the 14 datasets separated by method for the following metrics: `sepscore`, `avedist`, `klmetr`, and `pvca`. In each row the results are listed in descending order according to mean performance in terms of the original values and their ranks.

diffexpr

mean values	combat 0.11044	stand 0.10958	ratioa 0.10891	meanc 0.1088	ratiog 0.10796	none 0.10526	sva 0.09517	**fabatch** **0.09364**
mean ranks	combat 3.28571	stand 3.57143	ratioa 3.78571	meanc 4.14286	none 4.5	ratiog 4.64286	**fabatch** **5.85714**	sva 6.21429

skewdiv

mean values	**fabatch** **0.01724**	sva 0.02206	stand 0.02377	combat 0.02688	ratioa 0.02875	ratiog 0.03257	meanc 0.03671	none 0.05041
mean ranks	sva 2.21429	**fabatch** **2.92857**	combat 4.28571	stand 4.78571	meanc 5.07143	ratioa 5.42857	ratiog 5.5	none 5.78571

corbeaf

mean values	none 1	combat 0.86857	meanc 0.86742	ratioa 0.85516	ratiog 0.84931	stand 0.82754	sva 0.69313	**fabatch** **0.67795**
mean ranks	none 1	combat 2.92857	meanc 2.92857	ratiog 4.21429	ratioa 4.35714	stand 5.85714	sva 7.14286	**fabatch** **7.57143**

Table 3.3.: Means of the metric values and of the corresponding ranks among the various methods for the 14 datasets separated by method for the following metrics: `diffexpr`, `skewdiv` and `corbeaf`. In each row the results are listed in descending order according to mean performance in terms of the original values and their ranks.

two methods performed best (with the exception of FAbatch for Design B with common correlation). However, for the real data they performed worst—even worse than no batch effect adjustment in the mean. For FAbatch those datasets were examined which yielded substantially worse `diffexpr` values after batch effect adjustment than before. As can be seen in Figure 3.3, two of these datasets contain data with high `diffexpr` values before batch effect adjustment. This implies that for these datasets the biological signal is well preserved in the batches—in other words they seem to be less affected by batch effects. A possible reason FAbatch performs worse for mild batch effects has been outlined above. The other datasets connected with `diffexpr` values worse than "no batch effect adjustment" in the case of FAbatch were those for which some "outlying" batches were very different from the others—according to the PCA plots given in Figure 3.2. In this case, pooling the data of the outlying batch(es) with the other batches and estimating the L_2-penalized logistic regression model can result in a predictor with bad performance. The combined data might be too heterogeneous for the L_2-penalized logistic regression model, which assumes that all observations follow the same distribution. If the predictions of the class probabilities by the L_2-penalized logistic regression rule are bad, the biological signal is less protected in the latent factor estimation. Therefore, the removal of the estimated latent factor influences will affect the biological signal more. There were no noteworthy differences between the other methods with respect to `diffexpr`. For the real datasets none of the methods showed an advantage over no batch effect adjustment. This indicates that differential expression analysis might not benefit from batch effect adjustment in general.

For the skewness divergence score `skewdiv` (Figures B.12 and 3.3, Tables B.6 and 3.3) no clear ranking of the methods is seen in the case of the simulated data. However, for the real datasets, SVA and FAbatch clearly outperform the other methods with respect to this metric.

Finally, for both the simulated data and real data, FAbatch and SVA have considerably lower `corbeaf` values (Figures B.13 and 3.3, Tables B.7 and 3.3), which is not very surprising considering their high level of complexity.

3.3.2. Application in cross-batch prediction

In this illustrative analysis all batch effect adjustment methods outlined above were applied in cross-batch prediction together with the corresponding addon procedures described in section 3.2.3. A real data analysis as well as a simulation were performed. Luo et al. (2010) conducted a more extensive real data study. They used several datasets to compare all of the methods considered here, except for fSVA and FAbatch,

with respect to their performance in cross-batch prediction.

In the present analyis, the dataset `IUGRTranscr` was used because it features a relatively strong class signal and at the same time is strongly affected by batch effects, judging from the PCA plot in Figure 3.2. This dataset contains miRNA measurements obtained from 67 human placentas using the Illumina Human-6 v2 Expression BeadChip. Of these 67 samples, 27 were obtained from placentas of embryos suffering from intrauterine growth restriction (IUGR); the remaining 40 samples were obtained from placentas of healthy embryos. The dataset consists of one batch of 20 samples and another batch of 47 samples. In the first batch 9 (45%) samples and in the second batch 18 ($\approx 38\%$) samples originate from IUGR embryos.

As a classification algorithm for the dependent variable "IUGR (yes vs. no)" PLS-LDA was chosen, where the number of components used was tuned on the grid $1, 2, \ldots, 10$ employing 3-fold CV.

Just as in the extensive real data study of Luo et al. (2010), Matthews correlation coefficient (MCC) was used as a performance metric. This measure has an advantage over the more commonly considered misclassification error rate in that it is independent of the class frequencies in the test data. It takes values in $[-1, 1]$, where a MCC value of 1 would indicate a perfect prediction, a MCC value of 0 would correspond to a completely random prediction and a MCC value of -1 to a total disagreement between prediction and reality. The MCC is calculated as follows:

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}, \tag{3.26}$$

where TP designates the number of true positive predictions, TN the number of true negatives, FP the number of false positives, and FN the number of false negatives.

Figure 3.4 depicts the MCC values that result from applying the different batch effect adjustment methods in predicting from one batch to the other and then switching the training set and test set roles between the two batches. When training on the first batch, only ComBat, mean-centering, and FAbatch lead to a higher MCC value than does no batch effect adjustment. The two fSVA algorithms and standardization lead to a substantial deterioration in prediction performance, where the fast fSVA algorithm is slightly better than the exact fSVA algorithm. When training on the second batch, the prediction performance without batch effect adjustment corresponds to random guessing as indicated by the MCC value of zero here. Except for standardization and the exact fSVA algorithm, all methods lead to an improvement of prediction performance here. The ranking of the methods is almost entirely the same as that when training on the first batch.

In Figures 3.1 and 3.2 PCA plots were used to visualize batch effects in data after

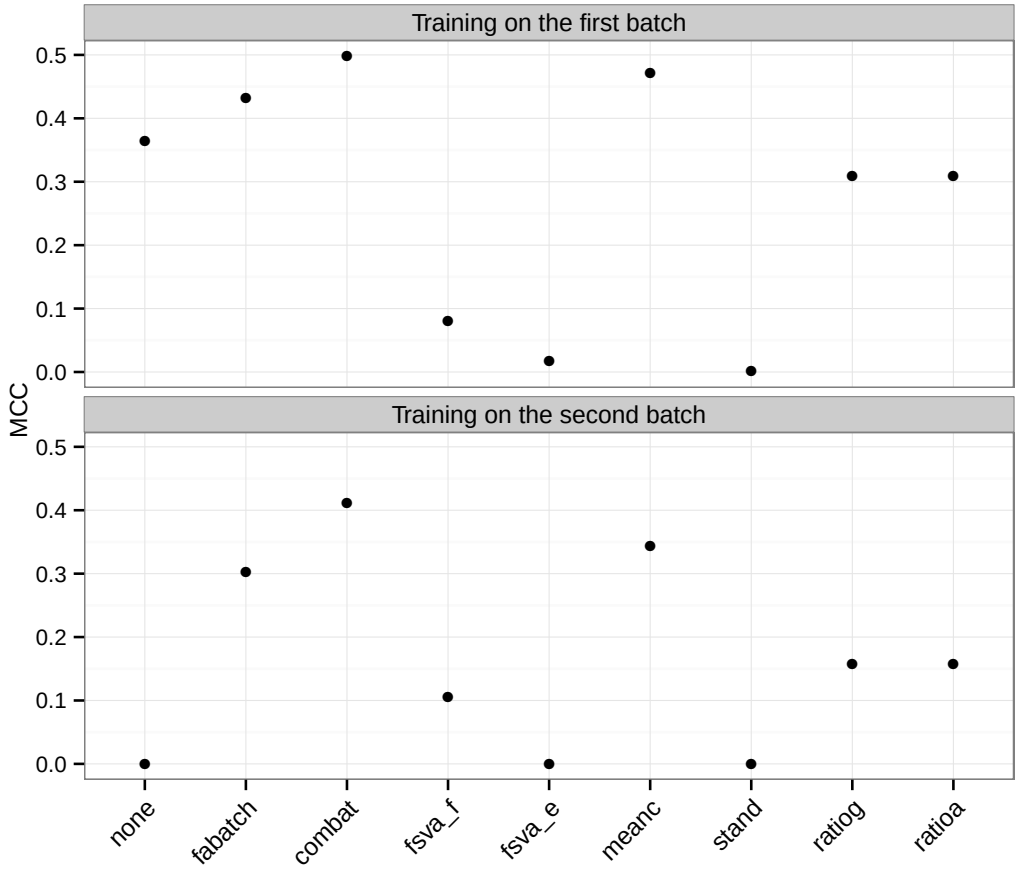

Figure 3.4.: Cross-batch prediction—MCC values. MCC values from using the individual batch effect adjustment methods in cross-batch prediction when training on the first and second batch. `fsva_f` and `fsva_e` denote the fast and the exact fSVA algorithm, respectively.

batch effect adjustment and in raw data, respectively. In this section such plots are utilized for a slightly different purpose: to study the extent to which the test batch is similar to the training batch after addon batch effect adjustment using the different batch effect adjustment methods. In each panel of Figure 3.5 the training batch is depicted in bold. In each case PCA was applied to the following data matrix: the training batch after batch effect adjustment combined with the test batch after addon batch effect adjustment using the method indicated in each case. The stronger the two point clouds overlap, the closer the test batch is to the training batch after addon batch effect adjustment. Before batch effect adjustment the two batches obviously are grossly disparate. While the shapes of the point clouds are rather similar, their locations differ considerably. FAbatch leads to the greatest overlap between the training and test batches, here. ComBat and standardization are in second place. Note that despite the decent overlap between training and test batches using standard-

ization, this method produced poor MCC values in the analysis above. In Chapter 4 it will be seen that, in general, when using PLS-LDA as a classification method, standardization leads to a poor prediction performance when employed to assimilate the validation data to the training data. Mean-centering, ratio-A, and ratio-G are connected with a worse overlap here and the point clouds hardly differ among these methods. The two fSVA algorithms make the two point clouds even more disparate than before batch effect adjustment. The poor performance of fSVA observed here indicates that in this example it seems inappropriate to assume that the same sources of heterogeneity operate in the two batches—an assumption required for the application of fSVA. In section 3.2.3 it was noted that for mean-centering, standardization, ratio-A, and ratio-G methods no specific addon batch effect adjustment methods are required because they treat each batch independently of the others. Therefore, for each of these methods, in the two corresponding subplots of Figure 3.5 the point clouds are identical, irrespective of which batch is used as the training batch and test batch.

Note again that the above real data analysis is illustrative only. Simulations give more accurate results and allow the study of the impact of specific aspects of the underlying data distribution. In the simulation presented in the following the main interest lied in demonstrating that FAbatch is best suited in situations with correlated predictors. Four simulation settings were considered: the three settings of Design B presented in section 3.2.4 and an additional setting in which no correlation among the predictors was induced. Design B was chosen instead of Design A in order to prevent a possible optimistic bias with respect to FAbatch and fSVA, since these involve adjustment for latent factor influences. The additional fourth setting was generated by simply setting the correlations in Design B to zero. For each setting 100 datasets were simulated and proceeded as in the analysis of the real dataset presented above—with two differences. The first difference was that in the simulation there were $\binom{4}{2} \times 2 = 12$ instead of two combinations of training batches and test batches per dataset, because the simulated datasets featured four batches instead of only two. The second difference concerns the evaluation of the results, because the MCC values could not be calculated in cases where the denominator in Eq. (3.26) was zero. Therefore for each combination of setting and batch effect adjustment method the TP, the TN, the FP, and the FN values were separately totaled up over all prediction iterations in all 100 datasets and the MCC value then was calculated using the standard formula. Figure 3.6 shows the results. In many respects the simulation results concur with the results obtained using the real dataset. The most striking difference is that standardization is best here although it was bad for the real data analysis. However, the good performance of standardization in the simulation

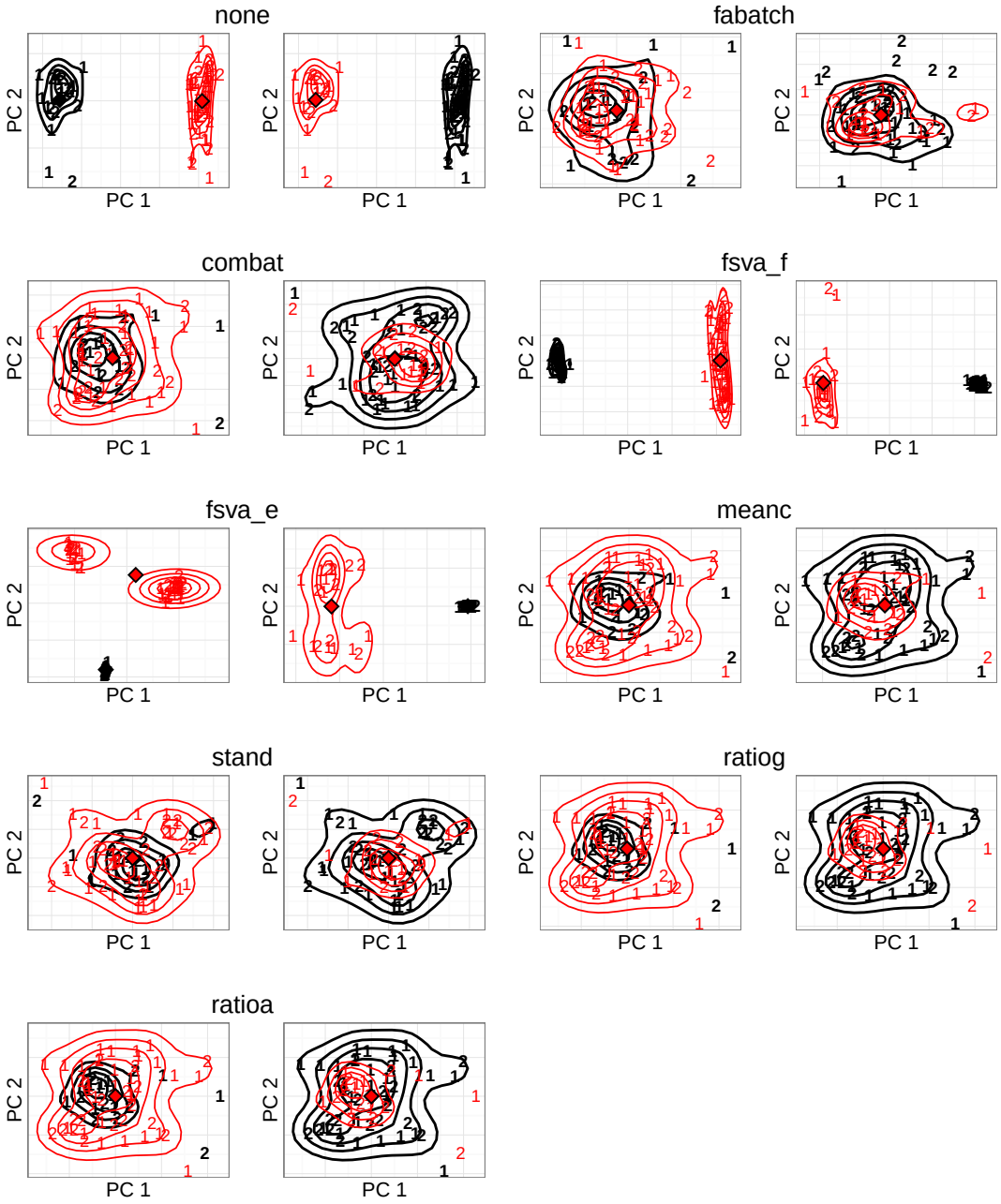

Figure 3.5.: Visualization of the assimilation of test batch to training batch after batch effect adjustment. First two principal components of PCA performed on the following data matrix: the training batch after batch effect adjustment combined with the test batch after addon batch effect adjustment. The training batch in each subplot is depicted in bold and the numbers distinguish the two classes "IUGR yes" (2) versus "IUGR no" (1). The contour lines represent batch-wise two-dimensional kernel estimates and the diamonds represent the batch-wise centers of gravities of the points.

should not be over-interpreted as it is the least performant method in the study of Luo et al. (2010). Moreover, as noted above, in Chapter 4 it is revealed that standardization performs poorly when used in combination with PLS-LDA. FAbatch is the second-best method in all settings except for the setting without correlation between the predictors. In the latter setting, FAbatch is outperformed by ComBat and mean-centering. This confirms that FAbatch is best suited in situations with more correlated variables. Ratio-G performs poorly here—other than in the study by Luo et al. (2010) and in the real-data analysis above. Both fSVA algorithms perform poorly here as well.

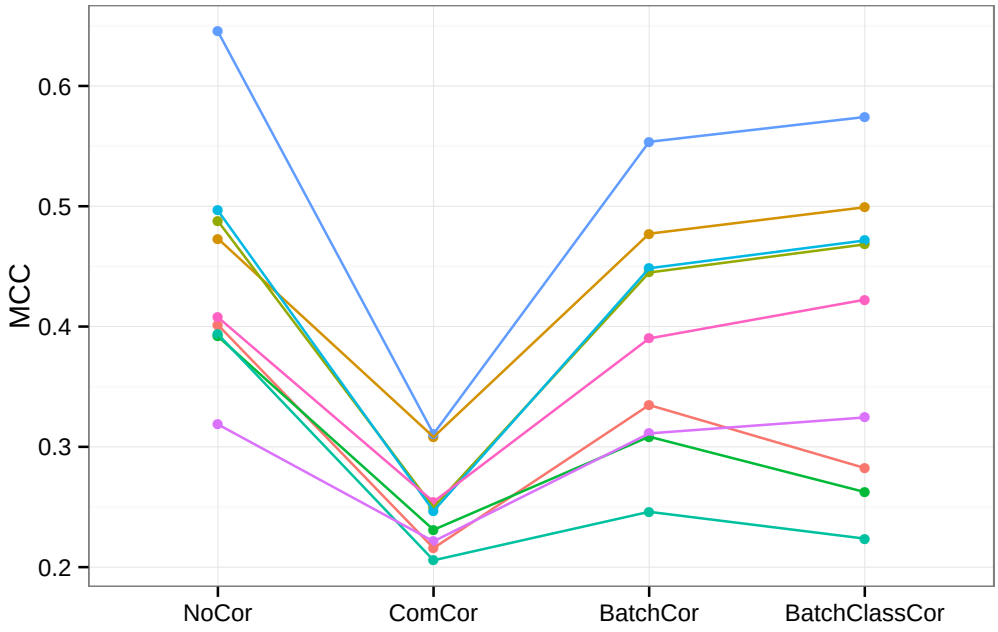

Figure 3.6.: MCC values from simulation study. The colors differentiate the methods: `none` (●), `fabatch` (●), `combat` (●), `fsva_f` (●), `fsva_e` (●), `meanc` (●), `stand` (●), `ratiog` (●), `ratioa` (●). For better interpretability the results corresponding to the same methods are connected.

3.3.3. Artificial increase in measured class signal by applying SVA

In section 3.2.2 it was explained in detail why using the actual values of the target variable in protecting the biological signal during the latent factor estimation of FAbatch would lead to an artificially increased class signal. SVA uses the values of the target variable and indeed suffers from the problem of an artificially increased class signal. In the following, the reason SVA suffers from this problem will be outlined. A serious problem with the weighting of the variable values by the estimated probabilities that the corresponding variable is associated with unmeasured confounders but

not with the target variable is the following: These estimated probabilities depend on the values of the target variable, particularly for smaller datasets. Naturally, due to the variability in the data, for some variables the two classes are, by chance, mixed to a much lesser degree. Such variables, for which the observed separation between the classes is greater than the actual—biologically motivated—separation, are connected with smaller estimated weights. This means that such variables are affected less by the removal of the estimated latent factor influences than are variables not connected with such a randomly increased separation. Phrased differently, the stronger the apparent—not the actual—signal of a variable is, the less its values are affected by the adjustment of latent factors. As a result, after applying SVA the classes are separated to a greater degree than they would be if biological differences among the classes were the only source of separation—as is required in a meaningful analysis. This phenomenon is pronounced more strongly in smaller datasets. The reason for this is that for larger datasets the measured signals of the variables approach the actual signals, which is why the overoptimism due to working with the apparent signals instead of the actual signals becomes less pronounced here. Accordingly, in the example with real data in the previous subsection fSVA performed considerably worse when using the smaller batch as training data.

Using datasets with artificially increased signals in analyses can lead to over-optimistic results, which can have serious consequences. For example, when the result of CV is over-optimistic, this may lead to overestimating the discriminatory power of a poor prediction rule. Another example is when searching for differentially expressed genes, where an artificially increased class signal could lead to an abundance of false positive results.

The observed deterioration of the MCC values in the real data example by performing fSVA when training on the smaller batch, admittedly, also may be due to random error. To determine whether the effects originating from the mechanism of artificially increasing the discriminative power of datasets by performing SVA are strong enough to have actual implications in data analysis, a small simulation study was conducted. Datasets with 40 observations were generated, each of which featured 1000 variables, two equally sized batches, standard normally distributed variable values, and a binary target variable with equal class probabilities. Note that there was no class signal in these data. Then, to each simulated dataset 5-fold CV repeated twice was applied to estimate the misclassification error rate of PLS-LDA. Consecutively, SVA was applied to the data and the misclassification error rates were estimated using the same procedure. This procedure was repeated for the following numbers of factors to estimate: 1, 2, and 3. In each case 50 datasets were simulated. The mean of the misclassification error rates was 0.504 for the raw datasets and 0.431, 0.356, and

0.306 after applying SVA with 1, 2, and 3 factors. These results confirm that the artificial increase in the class signal by performing SVA can be large enough to have implications in data analysis. Moreover, the problem seems to be more severe for a higher number of factors estimated. The same analysis also was done with FAbatch, again using 1, 2, and 3 factors, where the following misclassification error rates were obtained: 0.505, 0.521, and 0.509, respectively. The results indicate that FAbatch does not suffer from this problem in the context investigated.

3.4. Discussion

In this chapter, FAbatch, a general batch effect adjustment method, was introduced. It is applicable in situations where the batch membership is known and accounts for two kinds of batch effects simultaneously: 1) coarse, easily observable batch effects expressed as location and scale shifts of the variable values across the different batches; 2) more complicated batch effects, modelled by latent factor influences which affect the correlations among the variables in the batches. The model behind FAbatch is an extension of the model underlying ComBat, the latter of which is designed to address the first kind of batch effects described above. In FAbatch latent factors are used to model batch effects in the spirit of SVA. In contrast to SVA, however, FAbatch assumes that the batch membership of the observations is known and that the latent factor models are batch-specific, that is, that in each batch different sources of heterogeneity may operate. In section 3.2.1 it was shown that in the SVA model it is implicitly assumed that the distribution of the vector of latent factors may be different for each observation. This is a very broad assumption. However, it is unclear how well SVA can deal with specific datasets originating from such a general model because the link between the SVD used in the estimation and this model is not evident. By contrast, the estimation algorithm of FAbatch was motivated explicitly by its underlying model, which is quite general and reasonable. In cases in which the data in question are generally uniform with this model, FAbatch should perform reasonably well. In the form presented here, FAbatch is applicable in the presence of a binary target variable only. However, it also can be extended to other types of target variables. For example, when having a metric target variable, ridge regression could be used instead of L_2-penalized logistic regression when protecting the biological signal of interest in the factor estimation.

In an illustrative analysis the batch effect adjustment methods previously studied in the main analyses were applied in the important case of cross-batch prediction. FAbatch—unlike fSVA—performed reasonably well in this example. Moreover, by a

small simulation study evidence was obtained that the artificial increase in the measured biological signal of interest faced when performing SVA can have noticeable negative effects in applications. In FAbatch, this artificial increase is prevented by employing the following: For each observation the parameters involved in the transformations performed for protecting the biological signal are estimated using training data, which does not contain the respective observation to be transformed. This idea also may be applied to protect the biological signal of SVA, that is, when multiplying the variable values by the estimated probabilities that the corresponding variables are associated with unmeasured confounders but not with the binary variable representing the biological signal. More precisely, these probabilities could be estimated in a CV procedure—taking up, again, the idea used in FAbatch.

It can be dangerous to combine data from several studies to analyze a biological phenomenon of interest, independent of whether or not batch effect adjustment is used to assimilate the different datasets. In the context of clinical studies Shenhav et al. (2015) observed that often the biological signal of interest exists in only one of the investigated studies. The fact that there is signal in one of the studies considered can be due to the specific population under investigation in the respective study or to a particular study design. In such situations there is no signal in the population considered in the analysis. However, a strong signal in one of the individual studies can dominate the behavior of the data combined from several studies. Therefore, false positive results can be easily obtained when analyzing data from combined studies.

All batch effect adjustment methods considered in this chapter, together with the corresponding addon procedures and all metrics used in the comparisons of the methods, were implemented in or adopted into the new R package `bapred` available online from CRAN (Hornung and Causeur; 2015).

3.5. Conclusions

FAbatch leads to a good mix of observations across the batches, which is reassuring given the diversity of batch effect structures in real datasets. In the case of very weak batch effects and in the case of strongly outlying batches, the observed biological signal may be slightly altered by FAbatch. In an extensive comparison study of existing and new batch effect adjustment methods, no method was found to be best with respect to all metrics. Thus, it is difficult to formulate general recommendations: The choice of method may depend primarily on the goal of the researcher as reflected by the choice of the metric. Performing no batch effect correction at all is not recommended in any case.

This chapter was concerned mainly with situations in which batch effects are present in an available dataset. In such situations, batch effect adjustment can be performed prior to the analysis of interest to mitigate batch effects and, in consequence, to obtain more accurate and/or less biased results. The next chapter is concerned with a different situation: A specific dataset is used as training data for a prediction rule that is consecutively applied to varying external test datasets. In such a situation, batch effect adjustment can be employed differently, namely for adjusting the behavior of the test datasets to that of the training dataset by performing addon batch effect adjustment to improve prediction accuracy. Addon batch effect adjustment was described in section 3.2.3 and applied in section 3.3.2 in the context of cross-batch prediction.

4. Improving cross-study prediction through addon batch effect adjustment or addon normalization

4.1. Background

As seen in the previous chapters, a wide variety of modern classification methods can be used to construct prediction rules about the presence of diseases or disease outcomes of interest on the basis of high-dimensional, molecular data. Not only are such prediction rules used extensively in this thesis, but they very frequently appear in the literature. Although, to date, they seldom are applied in daily medical practice, potentially they could be established as useful tools to assist medical doctors in their decision making (van't Veer and Bernards; 2008). In addition to governmental policies, a major obstacle to broader application of such methods is batch effects, which lead to lack of comparability of patients' data needed for prediction, that is, the test data, to that the prediction rules are constructed on, the training data. High-dimensional bio-molecular measurements are highly sensitive to external conditions of the data generation procedure (Scheerer; 2009). Moreover, different datasets used to study the same biological phenomenon vary in terms of the study population. Thus, prediction rules can be expected to perform worse or considerably worse in practice than is suggested by the results of dataset-internal error estimation through CV (Castaldi et al.; 2011; Bernau et al.; 2014). Apart from the dissimilarity among datasets used to study the same biological signal of interest due to batch effects, an important reason for external validation is the following: Often while particular datasets feature a strong biological signal of interest, other datasets from the field of application do not (Shenhav et al.; 2015), as explained in section 3.4. Researchers may be inclined to build prediction rules using datasets that feature a strong biological signal precisely because of the fact that they do feature a strong signal. A prediction rule created using such a training dataset admittedly features a small CV error estimate, but performs poorly when applied to independent datasets in which

the strong signal found in the training dataset is not present. This poor performance on independent datasets would, however, go unnoticed without performing external validation. As stated in Chapter 1, the term *cross-study prediction* is used to refer to situations where a prediction rule is learned using data from a study and then is applied to independent external data from another study.

It is a desirable goal to reduce the error frequency of prediction rules applied in cross-study settings. Batch effect adjustment methods frequently are used to make the distributions of different datasets more similar not only within a study but also across studies. However, it is far less acknowledged that these methods also can be applied to make test data more similar to the training data in the context of prediction. In Chapter 3, specifically in section 3.2.3, this addon batch effect adjustment was discussed in detail.

Independent from addon batch effect adjustment, by normalizing the training data and test data simultaneously, the severity of batch effects would be greatly reduced. However, in the context of prediction the prediction rule must not depend on the test data. This condition would not be fulfilled when normalizing training data and test data together, because the training data would change each time new test data arrived. This pitfall is circumvented by addon normalization, which was discussed in Chapter 2 in a different context: Normalization of the training data is done without considering the test data. When normalizing observations in the test data, for those parameters of the normalization procedure which do not entirely depend on the individual samples, estimates obtained from the training data only are used.

In this chapter the potential improvement of cross-study prediction yielded by the use of addon normalization, addon batch effect adjustment, and the combination of these two is investigated through their application to 25 raw microarray datasets of the same chiptype. This large-scale neutral comparison study follows the recommendations made by Boulesteix et al. (2013) and Boulesteix (2013). Beyond the small illustrative (and often biased) real data studies provided in the great majority of papers presenting new methods, such neutral comparison studies yield crucial evidence to guide data analysis practice (Boulesteix; 2013; Gatto et al.; 2016). The large number of datasets considered in these studies increases the reliability of the conclusions substantially (Boulesteix et al.; 2015). In the study presented in this chapter, seven batch effect adjustment methods and the addon normalization procedure for RMA by Kostka and Spang (2008) are considered. The target variable considered for all datasets is "sex". CV delivers error rates close to zero here because the biological signal present in gene expression for explaining "sex" is very strong. However, the error rate estimated by cross-study validation will be seen to be much higher, although from a biological point of view, it should be possible to predict "sex" accurately based

on microarray gene expression data in cross-study settings as well. This illustrates that batch effects can deteriorate the accuracy of prediction considerably in this context and that CV does not reflect the true error rate to be expected when applying a prediction rule to an external dataset in practice.

Note that it is not meaningful to predict "sex" from a clinical point of view. However, for the purpose of the systematic large-scale study performed here it is important to analyze a large number of datasets with the same phenotype target variable and collected using the same chiptype, which was possible only for the target variable "sex". Despite the fact that the biological signal present in gene data explaining "sex" is very strong, "sex" can be seen as a substitute for a meaningful phenotype target variable. Moreover, "sex" has the advantage of being a clearly defined target variable. By contrast, for clinically relevant target variables it often is difficult to find several datasets featuring the same two biological groups, and definitions may be ambiguous. Keeping in mind that prediction performance is usually better for "sex" than for most other target variables, in the study presented in this chapter the absolute sizes of the performance measure values will not be examined; deliberate focus will be on the effect of addon batch effect adjustment and addon normalization.

Modern next generation sequencing (NGS) data is commonly associated with considerably reduced variability compared to microarray data (Bullard et al.; 2010), which is why batch effects should be weaker for NGS data. Nevertheless, as stated in Chapter 1, batch effects have been found to pose a problem for NGS data as well (Hansen and Irizarry; 2012). The question investigated in the study presented here is, thus, relevant beyond the special case of traditional microarray data.

Unlike in this study, where cross-study prediction is investigated, in the study by Luo et al. (2010) of addon batch effect adjustment cross-batch prediction within the same study is considered. In their paper, batches are parts of a common dataset which are uncomparable for reasons unrelated to the biological signal of interest. Since their batches originate from the same study, they share certain characteristics, for example, the laboratory used for data generation or the personnel involved may be the same for all batches. However, such similarities between training data and test data generally are not present in cross-study settings when a prediction rule is made publicly available and applied by other teams around the world. Therefore, the analysis design used in this study reflects practically relevant situations better. Moreover, by considering a large number of datasets, more stable results are obtained than in Luo et al. (2010).

This chapter is structured as follows: In section 4.2 a description of the data material is provided and the analyses performed in the cases of cross-study prediction using addon batch effect adjustment and addon normalization are explained. In sec-

tion 4.3 important features of the results of the large-scale comparison study are described. In section 4.4 several of the findings are interpreted and further opportunities for application of the methodology are proposed. In section 4.5 practically relevant conclusions from the chapter are drawn.

4.2. Methods

4.2.1. Data material

All datasets were obtained from ArrayExpress (Kolesnikov et al.; 2015). As a preliminary step datasets meeting the following criteria were sought: availability of a variable denoted as "sex" in the phenotypic data, availability of the raw data (necessary for (addon) normalization), number of samples between 30 to 500, human origin of the samples, and samples of microarray chip type Affymetrix HGU GeneChip HG-U133PLUS2. From the corresponding search results initially the 39 most recently published datasets meeting these criteria were considered. Subsequently, for each dataset an investigation was made into whether there were repeated measurements and if so, one sample per patient was chosen randomly. Following this, any datasets containing duplicates from other datasets were excluded. Moreover, datasets featuring fewer than 20 observations after removal of repeated measurements were excluded. After excluding further datasets seen to contain repeated measurements, ultimately 25 datasets for use in the analysis were obtained. Table 4.1 provides basic information on these datasets after removal of repeated measurements.

4.2.2. (Addon) Batch effect adjustment and (addon) quantile normalization

The seven batch effect adjustment methods investigated were the same as those considered in Chapter 3: ComBat, fSVA, mean-centering, standardization, ratio-A, ratio-G, and FAbatch. Both variants of fSVA presented in Parker et al. (2014) were considered: the exact fSVA algorithm and the fast fSVA algorithm (see also section 3.2.3). See Chapter 3 for a detailed discussion about addon batch effect adjustment. In the analysis presented in this chapter, all methods were used in exactly the same ways as in Chapter 3.

RMA normalization (Irizarry et al.; 2003) together with the addon quantile normalization procedure by Kostka and Spang (2008), which also was considered in Chapter 2, was used (see section 2.2.3 for a description).

Accession number	Number of observations	Proportion of female patients	Year of publication
E-GEOD-19722	46	0.33	2014
E-GEOD-28654	112	0.41	2015
E-GEOD-29623	65	0.38	2014
E-GEOD-39084	70	0.50	2014
E-GEOD-45216	31	0.19	2014
E-GEOD-45670	38	0.16	2014
E-GEOD-46474	40	0.35	2014
E-GEOD-48278	57	0.51	2015
E-GEOD-48350	58	0.52	2014
E-GEOD-48780	49	0.84	2014
E-GEOD-49243	73	0.48	2014
E-GEOD-50774	21	0.38	2014
E-GEOD-53224	53	0.60	2015
E-GEOD-53890	41	0.51	2014
E-GEOD-54543	30	0.27	2015
E-GEOD-54837	226	0.35	2014
E-GEOD-58697	124	0.64	2015
E-GEOD-59312	79	0.33	2014
E-GEOD-60028	24	0.67	2014
E-GEOD-61804	325	0.45	2014
E-GEOD-63626	63	0.62	2014
E-GEOD-64415	209	0.47	2015
E-GEOD-64857	81	0.42	2015
E-GEOD-67851	31	0.42	2015
E-GEOD-68720	97	0.44	2015

Table 4.1.: Overview of datasets used in the empirical study.

4.2.3. Cross-study validation

Bernau et al. (2014) recommend *cross-study validation* to obtain estimates of the error expected when applying prediction rules to external data. This procedure requires I datasets that study the same biological phenomenon. The prediction rule of interest is learned iteratively on each of the I datasets and its error evaluated on every other dataset. This results in $I(I-1)$ error estimates which are more realistic than CV error estimates as far as the application of prediction to external data in practice is concerned.

This procedure was slightly altered to fit the purposes of the study at hand. Instead of an error estimator a performance metric was considered, namely the MCC mentioned in section 3.3.2. The absolute size of the latter is interpretable analogously to that of the well known Bravais-Pearson correlation coefficient used with metric data. For this reason, it was favoured over the more common misclassification error rate. In the calculations, female patients and male patients are considered "positives" and "negatives", respectively. Again, as in section 3.3.2 the MCC values according to formula (3.26) would not have been calculable in cases where the denominator in the calculation of the MCC value was zero. Therefore, first, for each of the I training sets the TP, the TN, the FP, and the FN values were totaled up over the $I-1$ test set evaluations. Second, formula (3.26) was applied to the totaled TP, TN, FP, and FN values. Here, in some cases formula (3.26) was not applicable because the denominator was zero also in case of the totaled TP, TN, FP, and FN values. In each of these cases, the corresponding prediction rule either classified all observations as negative or all observations as positive so that $TP+FP$ or $TN+FN$ was zero. Such prediction rules, which simply assign all observations to one class, are no more effective than random guessing. Therefore, a MCC value of zero was assigned in these rare cases where either $TP+FP$ or $TN+FN$ was zero. The MCC values calculated using the totaled TP, TN, FP, and FN values are denoted as MCC_{rule}. This measure reflects the mean cross-study prediction performance of a specific prediction rule evaluated on test datasets from the setting under consideration.

4.2.4. Study design

Five parameters were varied in the analyses:

- normalization type: addon normalization (`addon`), separate normalization (`separate`)
- batch effect adjustment method: No batch effect adjustment (`none`), ComBat (`combat`), mean-centering (`meanc`), standardization (`stand`), ratio-G (`ratiog`),

ratio-A (`ratioa`), fast fSVA (`fsva_f`), exact fSVA (`fsva_e`), FAbatch (`fabatch`)

- Training set size: original size of dataset, but with a maximum of 70 observations (`trainlarge`), 20 observations (`trainsmall`)

- Test set size: original size of dataset, but with a maximum of 70 observations (`testlarge`), 20 observations (`testsmall`), 5 observations (`testverysmall`)

- Classification method: PLS-LDA (`PLS-LDA`), `PLS-LDA` using the 2000 variables with the smallest p-values from two-sample t-tests (`PLS-LDAvarsel`), Logit-Boost (`Boosting`), `Boosting` using the 2000 variables with the smallest p-values from two-sample t-tests (`Boostingvarsel`), NSC (`NSC`), RF (`RF`), kNN using the 2000 variables with the smallest p-values from two-sample t-tests (`kNNvarsel`)

After applying (addon) RMA normalization and (addon) batch effect adjustment to each training and test dataset pair, the data were used to build and apply the respective classifier. For kNN initial variable selection was performed because unlike the other classification methods used in the analysis, kNN classification does not weigh the variables by importance; its performance thus depends very much on the quality of the variables included (Pohjalainen et al.; 2015). All possible combinations of the values of these parameters were considered, leading to a total of 756 settings ($2 \times 9 \times 2 \times 3 \times 7$). In cases where subsetting was necessary, random samples were drawn from the datasets. Here, except in the case of `testverysmall`, it was ensured that the smaller class was represented by at least five observations. Because all possible pairs of training datasets and test datasets were considered, for each setting there were 25 MCC_{rule} values, each corresponding to a specific training dataset.

The R code written to produce and evaluate the results is available at the following link: `http://www.ibe.med.uni-muenchen.de/organisation/mitarbeiter/070_drittmittel/hornung/icsv_suppfiles/index.html`.

4.3. Results

Figures C.1 to C.7 show boxplots of the MCC_{rule} values for each classification method, separated by batch effect adjustment method, normalization type, training dataset size, and test dataset size. In the following, unless otherwise stated, the description of the results of the study is based on these plots.

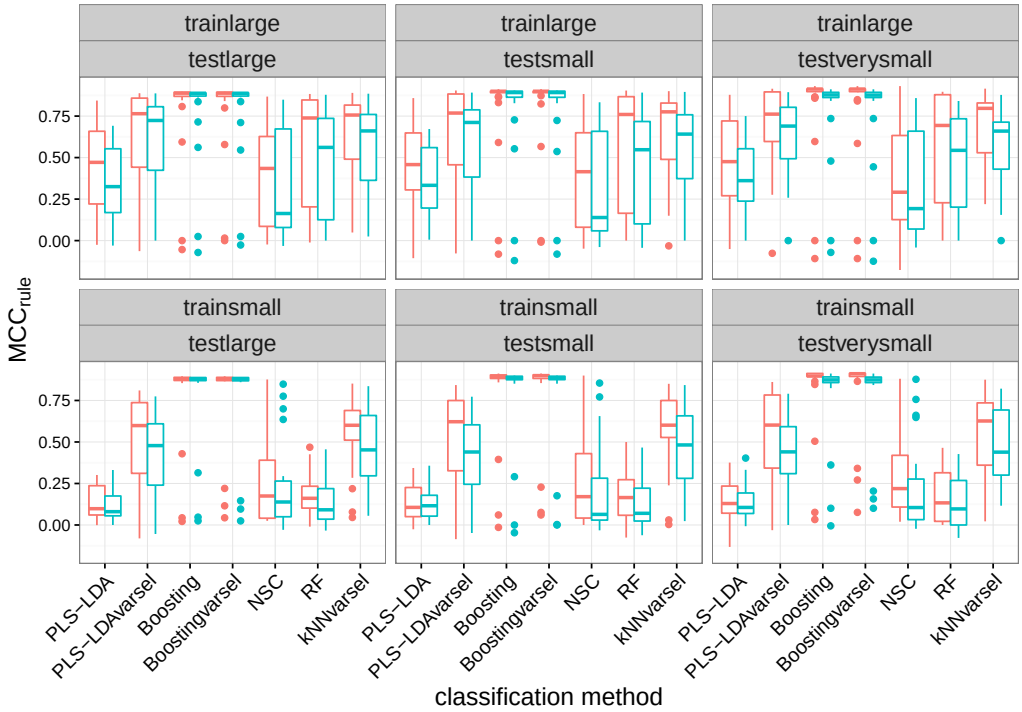

Figure 4.1.: MCC_{rule} values for the 25 datasets for each setting without addon batch effect adjustment. The red and the cyan boxplots indicate the results when using addon and separate normalization, respectively.

4.3.1. Addon quantile normalization

In most of the settings without addon batch effect adjustment, addon normalization improved performance and in no setting did it lead to a decline in classification performance (see Figure 4.1). While addon batch effect adjustment, if applicable, usually is more effective than addon normalization, in some situations it impairs performance (see further down). Because performance was not impaired by addon normalization in any of the settings studied, the following generally should be done: Addon normalization should be performed whenever test observations are unavailable in groups and addon batch effect adjustment is, thus, impossible, and it should be performed when addon batch effect adjustment tends not to improve results (see further down). While both approaches improve performance, there is no advantage to using addon batch effect adjustment in combination with addon normalization over using addon batch effect adjustment alone. Instead, in some cases the performance deteriorates slightly by additional addon normalization (see also section 4.4). Therefore, in the following, the results obtained for the combination of addon normalization and addon batch effect adjustment will not be examined; only the results obtained for either addon normalization or addon batch effect adjustment will be explored. Note that

addon quantile normalization is, however, not necessary in the case of rank-based classifiers, that is, classifiers which exclusively use the orderings of the variable values of the individual observations. The top scoring pairs classifier (Geman et al.; 2004; Tan et al.; 2005) is an example of such a method that has been found to perform comparably well to standard classification methods for high-dimensional molecular data (Geman et al.; 2008).

4.3.2. Addon batch effect adjustment

Influence of training set and test set size

As expected, the MCC_{rule} values tended to be smaller for the setting with smaller training datasets. A striking observation is that `RF` delivered useful predictions in the setting with larger training datasets only. Sonka et al. (2014) previously noted that RFs do not generalize well when using small datasets as training data. While the size of the training dataset does influence the cross-study prediction performance, it has almost no influence on the benefit yielded by addon normalization and addon batch effect adjustment. For the sake of clarity, the descriptions in the following will focus on the setting with large training datasets only.

Figure 4.2 shows the median MCC_{rule} values for all settings with large training datasets and separate normalization. Generally, there were hardly any differences in the results for addon batch effect adjustment when using a large dataset and when using a small test dataset. However, when using a very small test dataset (five observations), the MCC_{rule} values tended to become considerably smaller. This frequently led to a small deterioration from addon batch effect adjustment. Therefore, as a general rule, for addon batch effect adjustment to be effective very small test datasets should be avoided.

Specific classification methods

Given a test dataset comprising several observations, whether or not batch effect adjustment considerably improved results depended on the classification method used. For most classification methods an improvement through certain addon batch effect adjustment methods is observed (see the next subsection for details), the exceptions being `Boosting` (Figure C.3), `Boostingvarsel` (Figure C.4), and `RF` (Figure C.6).

In the case of `RF` the boxplots corresponding to `combat`, `meanc`, `stand`, `ratiog`, and `ratioa` have a very similar form. These methods all assimilate the means between the training data and the test data. Upon closer inspection of the results, the small 25% quartiles of the MCC_{rule} values displayed in the boxplots (Figure C.6) were found

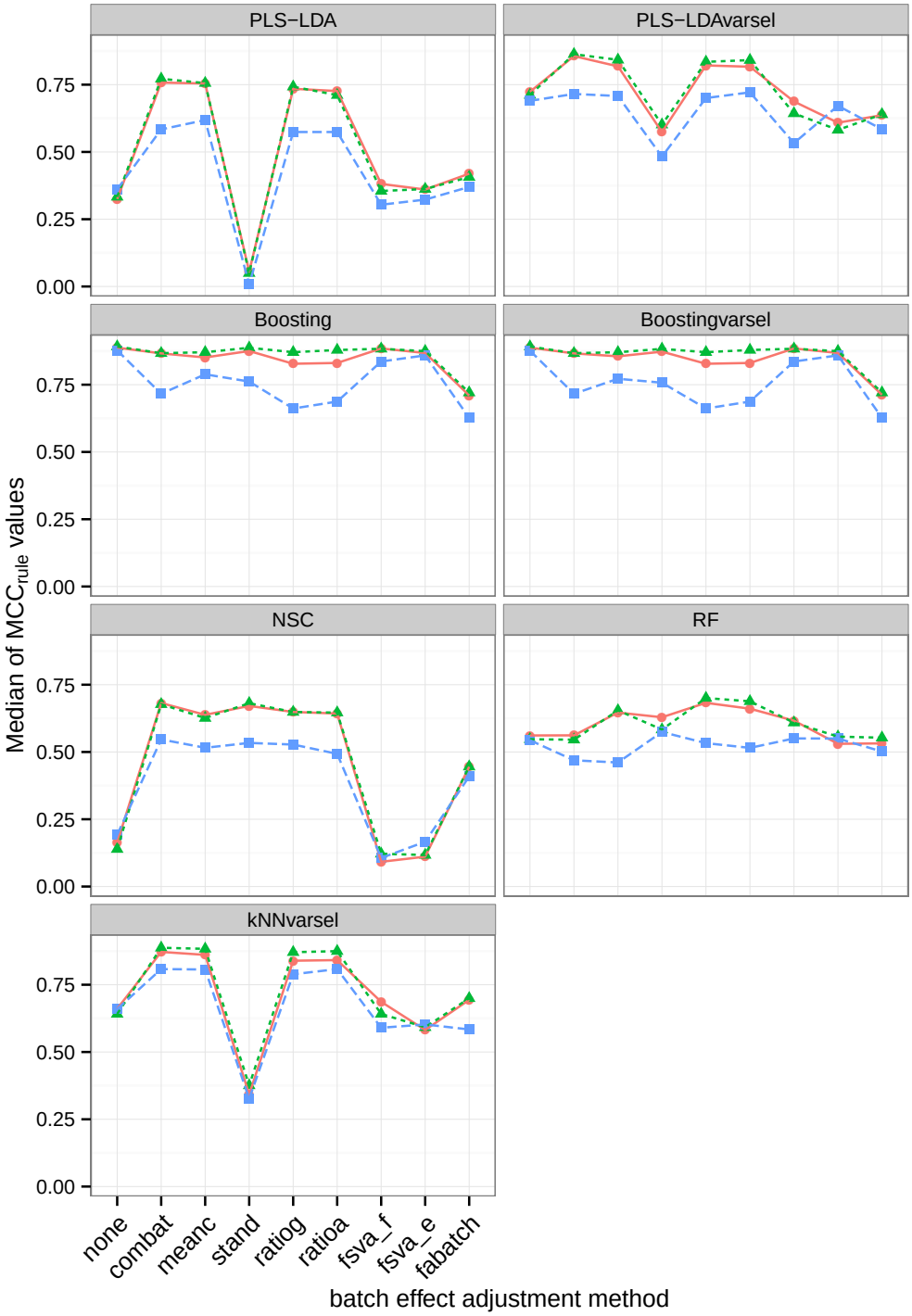

Figure 4.2.: Medians of the MCC_{rule} values over the 25 datasets for each setting with a large training dataset and separate normalization. The red lines, green lines, and blue lines indicate the results obtained when using a large, a small, and a very small test dataset, respectively.

to be attributable to the results of two training datasets, namely `E-GEOD-46474` and `E-GEOD-54543` (see Table 4.1). Here, the prediction accuracy was quite strong without addon batch effect adjustment. However, by applying the methods mentioned above, the results worsened to a substantially greater degree than was the case for other datasets for which deterioration was observed. These two problematic datasets were seen to be imbalanced with the frequencies of the smaller classes being 35% and 26.7%. For six of the remaining 23 datasets the frequency of the smaller class was below 35%. Five of these were associated with very small MCC_{rule} values not only with, but also without, addon batch effect adjustment. The fact that the performance was poor already without addon batch effect adjustment for the majority of imbalanced datasets explains why a substantial decline in performance through addon batch effect adjustment was observed for the two datasets mentioned but not for all imbalanced datasets. It is not surprising that RF performed poorly for many of the imbalanced datasets. RF is well known to predict overly frequently the class that is more frequent in the training dataset (see e.g. Janitza et al. (2013)). The deterioration in prediction performance through addon batch effect adjustment for the two datasets mentioned above is not directly due to the fact that in these cases the class frequencies are imbalanced in the training data. Instead, the reason is that the class frequencies in the test data tend to be very different from the class frequencies in the training data if the latter are imbalanced. The mechanism by which `RF` in particular suffers by differing class frequencies between training data and test data when used in combination with batch effect adjustment methods involving assimilation of the means between training data and test data will be explained in section 4.4.2. When excluding the two datasets for which substantial deterioration was observed, the boxplots showed a relatively strong improvement in the prediction accuracy of `RF` by addon batch effect adjustment (results not shown).

Both `Boosting` and `Boostingvarsel` performed very well without addon batch effect adjustment. Here, the MCC_{rule} values were very high and had almost zero variance apart from a few outliers (Figure 4.1; see the discussion section for an explanation why boosting may be especially suitable in cross-study prediction). Upon closer inspection of the results the small variance of the MCC_{rule} values observed for boosting without batch effect adjustment could be explained as follows: There were two to three datasets which performed poorly as training datasets and test datasets, while the other datasets exhibited an almost perfect performance. This had the effect that the MCC_{rule} values for the good training datasets were very similar because in these cases the totaled values used to calculate the MCC_{rule} values were almost the same: The corresponding prediction rules classified the observations from the good test datasets almost perfectly and those from the bad test datasets equally as badly.

The outliers in the lower domain mentioned above show the results obtained when using the bad datasets as training datasets. In conclusion, other than when considering the problematic datasets as training datasets or as test datasets, boosting without addon batch effect adjustment delivered almost perfect predictions. This explains why none of the batch effect adjustment methods led to an improvement here. For `combat`, `ratiog`, and `ratioa` there was substantial deterioration by batch effect adjustment for several training datasets, which seems surprising given that these methods performed well in other settings of the study (see next subsection). However, the training datasets for which substantial deterioration was observed for these methods were seen to feature, unexceptionally, the greatest degrees of class imbalance. Thus it can be concluded that boosting, as `RF`, can suffer from differing class distributions between training data and test data when used in combination with addon batch effect adjustment. Another factor that could contribute to the worse results for addon batch effect adjustment observed here is the variability that is associated with batch effect adjustment. This variability may be responsible for some differences in the predictions compared to when no addon batch effect adjustment is used. Such changes necessarily lead to errors when the predictions are almost perfect, as was the case for boosting.

Apart from the level of class imbalance in the training datasets, another factor that might influence the degree of improvement obtained through addon batch effect adjustment is the level of heterogeneity of the observations in the training data. To study whether the degree of heterogeneity in the training data is indeed a relevant factor, it was proceeded as follows. First, PCA was applied to each of the 25 datasets and in each case the first two principal components plotted against each other. Using these plots the datasets that featured a considerable level of heterogeneity were identified. Second, the results of the study obtained when using these heterogeneous datasets as training datasets were compared to the corresponding results obtained using the remaining datasets for training. Here, there were no clear indications that prediction rules obtained using heterogeneous training datasets may benefit less from addon batch effect adjustment.

For boosting, pre-selection of influencing variables performed with `Boostingvarsel` did not further improve results (Figure 4.1). By contrast, PLS-LDA seems to be improved by initial supervised variable selection, which also was found by Li et al. (2007).

Performance of individual batch effect adjustment methods

As seen above, the settings in which addon batch effect adjustment was not valuable were marked down. In those settings where it did improve performance there were several well performing methods with no clear ranking among them (Figure 4.2). Four methods always were among the best here: `combat`, `meanc`, `ratiog`, and `ratioa`. While `stand` also was frequently among the best methods, it was inferior in the cases of `PLS-LDA`, `PLS-LDAvarsel`, and `kNNvarsel`. Thus, the value of this method depends heavily on the classifier used, which is why it cannot be recommended. In contrast to the findings of Luo et al. (2010) in the present study `ratioa` and `ratiog` were not found to be preferable over the other well performing methods. `fsva_f`, `fsva_e`, and `fabatch` did not improve performance in any of the settings and, more importantly, these methods often were harmful and therefore should not be used for cross-study prediction. Note again that as explained in section 3.2.3 fSVA relies on similarity between training data and test data, an assumption most often not given in cross-study prediction. Results of the present study indicate that fSVA can impair performance when the assumption of similarity cannot be made.

The results of the illustrative real-data cross-batch prediction analysis presented in section 3.3.2 differed in the following ways: `fabatch` was among the best methods; `ratiog` and `ratioa` were not. Apart from the fact that in cross-batch prediction the training datasets and test datasets are more similar than in cross-study prediction, the differing results are likely due to the fact that in the illustrative cross-batch prediction analysis only a single dataset was used. In the cross-batch prediction simulation performed and reported on in section 3.3.2 FAbatch was, again, among the best methods. However, the simulation design used there was based on two real datasets. Therefore, the simulation results may as well depend in part on the behavior of these datasets used in designing the simulation.

4.4. Discussion

4.4.1. Reasons for no benefit from combining the two approaches

Used separately, addon normalization and addon batch effect adjustment both improved the performance of cross-study prediction under the conditions outlined in the previous section. However, there was no additional gain in prediction performance by using addon batch effect adjustment in combination with addon normalization in comparison to using addon batch effect adjustment alone. Two explanations for this could be the following: 1) The assimilation of the distribution of the test data to that

of the training data by addon batch effect adjustment is not substantially improved by a preceding addon normalization. Generally, addon batch effect adjustment leads to a stronger assimilation of the distribution of the test data to that of the training data than addon normalization. The reason for this is that addon batch effect adjustment explicitly assimilates the distributions of the individual variables in the test data to those in the training data. By contrast, addon normalization merely assimilates the marginal distributions of the values belonging to the individual observations. The latter is, however, implicitly performed by addon batch effect adjustment as well; 2) the variability connected with the adjustment is increased by combining the two procedures.

4.4.2. Random forest: impaired performance in the presence of differing class frequencies between training data and test data

When the classes were imbalanced in the training data, the performance of `RF` was impaired to the same extent by all addon batch effect adjustment methods, which involve an assimilation of the variable means between training data and test data. This was attributed to the fact that if classes are imbalanced in the training data, there tends to be a difference in class frequencies between the test data and the training data. In the context of conventional batch effect adjustment, Nygaard and Rødland (2016) noted that mean-centering reduces class differences when the classes are unevenly represented in the different batches. While all classifiers can be expected to suffer to some extent from variable mean adjustment, if there is a difference in class frequencies between training data and test data, this is probably particularly problematic for RFs. In the following the mechanism responsible for this will be described. The classification trees constituting a RF iteratively divide the observations into subgroups of decreasing sizes. More precisely, in each iteration the subgroups are split into two smaller subgroups based on a threshold of an individual variable. That threshold among all possible thresholds of the variables (in the randomly chosen subset) is used that leads to the strongest separation of the two classes through the two resulting subgroups according to a specific criterion. As a result, the splits are performed in each case using the variable (from the candidates) that has the greatest discriminatory power. The stronger the discriminatory power of a variable, the greater it suffers from an adjustment of the means between training data and test data if the class frequencies of the two are different. Here, the mean adjustment leads to the split point in the test data, which actually is the best, meaning that which separates the two classes in the test observations best, being strongly shifted away

from the best split point in the training data. The best split points in the test data are always shifted into the direction of the same class, namely that which is more frequent in the training data than in the test data. Thus, when splitting the test observations according to the split points found in the training data, many of the test observations belonging to the class less frequent in the training data are placed into the wrong subnodes. These wrong decisions accumulate as the test observations reach lower layers of the classification trees. In the extreme case, the RF ultimately classifies all test observations as the class which is more frequent in the training data than in the test data. For the two problematic training datasets mentioned above, it was investigated whether this phenomenon can be observed in the case of ComBat. Here, indeed ComBat led to classifying almost all test observations as the class over-represented in the training data. This was not the case without addon batch effect adjustment.

4.4.3. Boosting as a (potentially) robust method to avoid overfitting in the context of cross-study prediction

As seen in section 4.3.2 Boosting without batch effect adjustment almost perfectly predicted the class values across datasets. It has been noted in the literature that boosting is quite resistant to overfitting, that is, to an over-adjustment to the training dataset, in classification settings in particular (Bühlmann and Hothorn; 2007). While LogitBoost can be prone to overfitting, this can be efficiently inhibited by stopping the boosting iterations early (Bühlmann and Yu; 2008), as performed in this study.

Conventionally the term *overfitting* refers to the phenomenon that a classifier is overly adjusted to the specific observations in the training data. This can have the effect that the classifier features an increased error frequency when applied to independent test observations following the same distribution as the training data. In the context of cross-study prediction, however, independent test observations follow a distribution different from that of the training data, which is due to batch effects, as already mentioned. Therefore, a different kind of overfitting has to be considered here. A classifier may be overly adjusted not only to the specific observations in the training data, but also to the distribution of the training data. Such a classifier, which is adjusted too much to the particular behavior of the training data, may feature a poor generalizability to different, albeit similar, data distributions. A classifier of this kind would have a low level of CV error but a high level of cross-study prediction error. By contrast a classifier which does not overfit the training data distribution could have quite a high level of CV error but a low level of cross-study prediction error. Accordingly, Bernau et al. (2014) found only a weak positive correlation between CV

and cross-study validation error in their study.

The strong performance of boosting with early stopping suggests that this method may be resistant not only to overfitting the training observations, but also to overfitting the distribution of the training observations. Early stopping of the boosting iterations has the effect that only strong, coarse properties of the relationship between covariates and response in the training data are taken into account. These properties can be expected not to be induced by batch effects but to be common to all datasets on the biological phenomenon of interest. As the number of boosting iterations increases, the classifier is increasingly well adjusted to the training data distribution. This, together with the fact that boosting is more prone to overfitting for other prediction settings than classification could explain why the CoxBoost algorithm was less suitable for cross-study prediction in the study by Bernau et al. (2014) than LogitBoost was here. Similar to the number of iterations in boosting, other classification methods also feature tuning parameters which control the degree to which the algorithm adjusts itself to the training observations and consequently also to the distribution of the training data. Examples include the shrinkage parameter Δ of NSC and the penalization parameter λ in L_1- and L_2-penalized logistic regression. Further research could focus on the influence of such parameters on the cross-study prediction performance of these methods. The number of iterations in boosting could be especially useful in this context. First, this parameter has been seen to influence the performance considerably (see e.g. Seibold et al. (2016)). Second, in each iteration the influence of only one variable is updated, which is why boosting is not heavily dependent on the specific correlation structure of the dataset. Instead, new variables are consecutively taken into the model based on their importance with respect to explaining the target variable, and the iterations are stopped as soon as the model is deemed complex enough.

4.4.4. Further possibilities for application

ComBat holds a special place among the four well performing batch effect adjustment methods because of the peculiarity that the training data are not altered in any way by the adjustment. As a consequence, ComBat addon adjustment could be employed to improve the prediction performance of already existing prediction rules provided the following requirements are met: The training data used to learn the prediction rule must be available and the observations to predict must be available in groups of sufficient sizes.

In the analysis performed in this study quantile normalization was considered as part of RMA for Affymetrix data. However, quantile normalization also is used for

many other biomolecular data types (Okoniewski and Miller; 2008; Schmid et al.; 2010; Bullard et al.; 2010; Staaf et al.; 2008; 't Hoen et al.; 2008). Therefore, addon quantile normalization can be used for data types other than Affymetrix data to improve the cross-study performance of prediction rules obtained from these data types. Note that quantile normalization can be used with normalized data as well. This is important in view of clinical settings, in which raw data are not available on a standard basis.

4.5. Conclusions

Assimilating the test data to the training data before applying prediction rules obtained from gene expression data can improve the accuracy of prediction considerably. In this endeavor, both addon normalization and addon batch effect adjustment are in principle recommendable, but not the combination of these two approaches. Nevertheless, there are two requirements for the application of addon batch effect adjustment: 1) the test observations are available in groups of sufficient sizes; 2) the class frequencies must not differ strongly between training data and test data. If these requirements are met, addon batch effect adjustment with an appropriate method is preferable to addon normalization. The following addon batch effect adjustment methods are recommended and perform comparably well: `combat`, `meanc`, `ratiog`, and `ratioa`. All methods for assimilating training data and test data applied in the study are available in the R package `bapred`, version 1.0 (Hornung and Causeur; 2016), available from CRAN.

5. Conclusions and outlook

This thesis addressed specific aspects of preparation of high-dimensional biomedical data from a statistical point of view. Such focus on data preparation seems relatively unusual in (bio-)statistical research, which usually is immediately concerned with methodology for high-level analyses, which directly tackle specific questions of interest in the field of application. In the following, the most important conclusions drawn in Chapters 2 to 4 will be summarized, additional points within the context of this thesis not covered in the chapters will be discussed, and suggestions for future research on the topics explored in the thesis will be made.

Chapter 2 was concerned with investigating the reduction of the error estimated by CV procedures through performing data preparation steps before CV instead of including them in CV. Accordingly, the new CVIIM measure was defined as the relative reduction in the expected error obtained by CV by performing a data preparation step under consideration before CV. As the extent of underestimation by incomplete CV is not the same for each dataset, CVIIM depends on the underlying data distribution. With the global CVIIM a measure was introduced, which in contrast, does not depend on the underlying data distribution, but rather only on the specific data preparation step and potentially on the specific design of the analysis used in constructing the prediction rule. However, naturally it is dangerous to rely on global CVIIM alone, because without considering CVIIM for individual datasets, important situations where incomplete CV does lead to a relevant underestimation of the error of prediction rules can be overlooked. The plug-in estimator of CVIIM features a negligible bias but a relatively large variance, which is smaller for smaller CVIIM values.

With the aid of CVIIM and large collections of real biomolecular datasets, the following conclusions could be drawn in Chapter 2: RMA and RMAglobalVSN can be performed before CV using the entire dataset, but PCA should be included into CV through its addon procedure. If PCA is performed on the entire dataset before CV, there is a risk of severe underestimation of the misclassification error. More precisely, there is a risk of underestimating the misclassification error expected when applying the corresponding prediction rule to test data following the same distribution as the training data using the addon procedure for PCA. Preliminary results were obtained

for the following steps: CV-based optimization of tuning parameters, variable filtering by variance, and k-Nearest Neighbors imputation. These results are considered preliminary because only a limited number of datasets were used to obtain them. To draw valid conclusions from real data analyses about whether excluding specific steps from CV can be warranted, larger collections of datasets are required, which might deter researchers from studying the impact of a step under consideration. However, depending on the step of interest it may be meaningful to use simulated data instead of real datasets because simulated data may reduce the effort associated with studies in which the impact of data preparation steps concerning CV incompleteness is evaluated. Here, it would be important to set up several reasonably realistic simulation settings in order to ensure that common situations leading to large CVIIM values for the step under investigation are not overlooked. Data preparation steps for which the simulation approach would not be appropriate are, for example, normalization, because raw microarray data are difficult to simulate (see Nykter et al. (2006) for a sophisticated simulation procedure for this data type) and imputation, because the missing data mechanism is unknown in applications.

Throughout the thesis it was assumed that the prediction rule must remain fixed when new test data arrive. Under this presumption it is not possible to include the test data when performing unsupervised data preparation steps, that is, steps that do not take the target variable into account. However, for unsupervised steps that tend to feature high CVIIM values, the $e_{incompl,K}(\boldsymbol{s})$ values by definition tend to be much lower than the $e_{full,K}(\boldsymbol{s})$ values, which is why the prediction error would be greatly reduced by including the test data before performing the respective steps.

In general, unsupervised data preparation steps originally were not intended to be used in combination with addon procedures for the purpose of including new test observations. For some of these steps, the estimates of the parameters necessary for addon preparation may depend too much on the specific set of observations in the training data, which is why they may not be well suited for independent observations. This would explain high CVIIM values for this category of steps. Depending on the gain in prediction performance to be expected by performing a specific step under the inclusion of the test data, it might be worthwhile sometimes to refrain from the requirement of fixing the prediction rule.

A possible explanation why, in the specific case of PCA, large CVIIM values frequently were obtained is given in the following. With PCA, a separate loading of each variable is estimated for each component. Therefore, from a statistical point of view, many parameters have to be estimated for each component in the case of high-dimensional datasets. In their entirety, the estimates of these parameters depend heavily on the training dataset used. This is similar to the overfitting of statistical

models for explaining a target variable in the presence of high-dimensional covariate data, occurring, for example when using penalization parameters that are too small in penalized regression approaches. As a consequence of considering such a multitude of parameters for each component, the factor loadings in their entirety are suited much better for an observation that is added to the training data before performing PCA, in particular for heterogeneous data. This would explain why, for several datasets, high CVIIM values were obtained, but for the majority of datasets only small values were obtained.

Note that also in the case of fSVA a multitude of parameters have to be estimated. This fact, as well as the fact that the training datasets and test datasets behave quite differently, may contribute to the poor performance of fSVA observed in cross study-prediction. In a potential, sparse version of SVA, a smaller number of parameters would have to be estimated. Therefore, the cross-batch prediction performance might be improved by such a modification to the method.

In all analyses of the impact of CV incompleteness with respect to specific steps and in the definition of CVIIM, the target variable was binary. The impact of CV incompleteness may be stronger for metric target variables. In general, the information contained in a metric variable is more detailed than that in a binary target variable. Therefore, a prediction rule for a metric variable generally could be better adjusted to the training data than a respective prediction rule for a binary variable. Consequently, a prediction rule for a metric target variable could be more sensitive to the impact of CV incompleteness, which would lead to higher CVIIM values than in the case of a comparable prediction rule for a binary target variable. However, this is merely a presumption that was not in any way empirically investigated.

As stated in Chapter 4 batch effects are weaker for modern NGS data. Because modern biomolecular measurements are becoming increasingly accurate and studies increasingly homogeneous through improved coordination in the information age, batch effects will become less important in the future. Therefore, from a technical point of view, in more and more applications it may become less crucial to validate prediction rules externally before using them in practice. As a consequence, in many cases only CV error estimates might be reported instead of error estimates obtained using external data.

When relying on CV alone for error estimation, it is especially important that this procedure be conducted in such a way that it is not connected with a relevant optimistic bias. Here, it will be valuable to perform empirical studies using CVIIM to ensure that no data preparation steps, which lead to a strong underestimation of the prediction error, are performed on the entire dataset before CV.

Another consequence of a decrease in batch effects would be that addon procedures

for data preparation steps play a more important role. This is because the application of addon procedures made necessary by performing corresponding data preparation steps in the construction of prediction rules, in general, requires the test data to follow the same distribution as the training data. Given the expected increase in importance of addon procedures, it is worthwhile to derive and implement the addon procedures for several important data preparation steps. Availability of these procedures would allow researchers to integrate the data preparation steps used in their analyses into their prediction rules and present the latter, for example, in the form of a Shiny-based user interface available online. Here, users of the prediction rules could simply upload the raw data of new patients and automatically obtain predictions. These practical aspects of integrating data preparation steps into prediction rules are discussed in an upcoming paper that I am co-authoring (Boulesteix et al.; in prep).

The new FAbatch method introduced in Chapter 3 extends location-and-scale adjustment of the batches as performed with ComBat by adjustment for latent factor influences within batches. Figure 3.5 on page 79 shows that FAbatch—in contrast to fSVA—performs well in assimilating the distribution of the test data to that of the training data. However, while FAbatch performed quite well in cross-batch prediction, it impaired performance for cross-study prediction. It is not certain why this is the case. A crucial point is that in the latent factor estimation with FAbatch the biological signal is not entirely protected, neither in the training data nor in the test data. It could be worthwhile to compare FAbatch thoroughly with a mere location-and-scale adjustment of the batches and to inspect more closely those test dataset observations that are correctly predicted with location-and-scale adjustment but falsely with FAbatch. Thereby specific properties of test observations that are difficult to classify when using FAbatch could be revealed and FAbatch could be adjusted accordingly to improve its performance with respect to classifying these observations.

In protecting the biological signal in the training data, with FAbatch rather than with fSVA the actual classes are not used; instead, class probabilities are estimated through CV. The reasons for this are the following: 1) If the actual classes were used, the biological signal would be exaggerated in the training data; 2) In the adjustment of the test data, the actual classes cannot be used because they are unknown; and 3) Using the actual classes in the training data, but not in the test data, would not correspond to an addon batch effect adjustment procedure as defined in section 3.2.3. An advantage of using the actual classes in the training data would be that there would be no risk of the biological signal in the training data being diminished by the adjustment for latent factors. However, as noted above, if the actual classes are used, the signal is exaggerated. One might investigate whether the latter disadvantage of using the actual classes in the training data is outweighed by the former advantage in

the sense that a better cross-study prediction performance might be achieved when using the actual classes. This is contradicted, however, by the fact that FAbatch performed poorly in the real data analysis presented in Chapter 4 even though in this analysis the biological signal was very strong and the respective estimated class probabilities in the training datasets were consecutively close to zero and one.

The large real data study performed and reported on in Chapter 4 revealed that under some conditions, cross-study prediction can, indeed, be improved through certain addon batch effect adjustment procedures and addon normalization but not through the combination of these two approaches. It is questionable as to whether a batch effect adjustment method can be developed which performs considerably better in cross-study prediction than the four best-performing methods discussed in Chapter 4—`combat`, `meanc`, `ratiog`, and `ratioa`. A new, considerably better batch effect adjustment method may not be possible, because although the approaches underlying these best-performing methods are reasonably different, their median performances are surprisingly similar. On the other hand, there is nonetheless room for improvement.

For `stand` performance depended heavily on the classification method used: When using `PLS-LDA`, `PLS-LDAvarsel`, and `kNNvarsel` this method performed considerably worse than for the other classifiers, where it was among the best methods. An important reason for the poor performance of `PLS-LDA`, `PLS-LDAvarsel`, and `kNNvarsel` when used in combination with standardization could be that they probably favor variables with a high level of variance when predicting new observations. Highly variable genes are known to be frequently associated with phenotypes (Li et al.; 2010; Alemu et al.; 2014), thus these are good candidates for differentially expressed genes, in particular with respect to a generic target variable such as "sex", which was considered in the analysis in Chapter 4. The euclidean distances between observations that are considered in the kNN algorithm are dominated by variables with high variances, which is why such variables influence the choice of the k nearest neighbors more than variables with low variance. Due to regression towards the mean, variables with a high level of empirical variance in the training data are likely to have a lower level of empirical variance in the test data. This in turn has the effect that the estimated loadings used to construct the PLS components in PLS-LDA are overly large for the values in the test data in the case of highly variable genes. Consequently, these genes play a more dominant role when predicting the values of the components in the test data than do genes with small variances.

In Chapter 4 it was seen that if in cross-study prediction the class frequencies in the test data differ substantially from those in the training data, addon batch effect adjustment can impair the performance substantially. In these cases the variable

values in the test data are shifted too far into the direction of the class that is more frequent in the training data than in the test data. In practice, the training data often are more balanced than the test data are. This is because, in general, the incidence of the disease under consideration is higher in the study from which the prediction rules are constructed than in the population to which they are intended to be applied. In many situations in practice the approximate percentage of individuals affected by a specific disease is known. This information could be used in the assimilation of the test data to the training data. For example, for adjusting the variable means in the test data to that of the training data, the following could be done: 1) Center the variable values in the test data to have zero means; 2) Instead of adding the variable means of the training data to the values resulting from 1) add to these values for each variable the weighted mean of the two class-specific variable means of the training data with weights proportional to the proportions of individuals of the respective classes in the population from which the test data originates. This would prevent the variable values in the test data from being shifted too far in the direction of the class more frequent in the training data than in the test data.

Not only was there no single best performing addon batch effect adjustment method in the case of cross-study prediction, also in the more general comparisons of the batch effect adjustment methods discussed in Chapter 3, there was no method that was best in terms of all aspects investigated. This suggests that an omnipotent batch effect adjustment method performing well in all respects may not exist. FAbatch was particularly effective for mixing together observations from different batches, but failed to preserve the biological signal of interest when there were extremely outlying batches and when the batch effects were very weak compared to the biological signal.

Although none of the batch effect adjustment methods studied was omnipotent, the methods differ with respect to different aspects of measuring the performance of batch effect adjustment. Therefore, it could be fruitful to develop methods targeted to individual analysis goals. For example, focus could be on developing a batch effect adjustment method that is particularly efficient with respect to the identification of influential variables in multiple testing.

A potential method in this context could be based on the ComBat algorithm. Although, just as the other methods considered, ComBat lead to no improvement in the `diffexpr` values for the real datasets it was best with respect to `pvca` here (see Figure 3.3). This suggests that ComBat is particularly effective with regard to preserving the biological signal of interest. Although the aim of standardization just as that of ComBat is to assimilate the means and variances of the variables across batches, the `pvca` values for standardization were smaller than those obtained in the case of ComBat. This suggests that the reason for the good performance of ComBat observed

here, lies in the empirical Bayes-based shrinkage this method performs. More precisely, with ComBat the empirical means and variances within batches are not made exactly equal to, but only shrunken towards, the overall means and variances (see Johnson et al. (2007) for details). As a consequence, larger deviations in the batches in terms of means and variances, which might not be explainable entirely by batch effects but might be due to biologically relevant differences among the batches, are only partially removed. This mechanism is probably responsible for the better preservation of the biological signal with ComBat than with the other methods. The degree of shrinkage performed by ComBat is determined by the empirical Bayes procedure. However, it might be possible to choose a degree of shrinkage that is particularly effective for the application of multiple testing, for example one that leads to a very small false discovery rate.

An interesting aspect related to prediction, but not to data preparation, that was touched upon in section 4.4.3 is the potential improvement in cross-study prediction by choosing different values of important tuning parameters of conventional classification methods. In classic settings, where it is assumed that the test data follow the same distribution as the training data, the purpose of tuning parameters is to inhibit an overly strong or even a perfect adaption of the respective model to the specific observations constituting the training data. Naturally, tuning also fulfills this function in cross-study prediction—in some cases the corresponding models would not even be estimable without the constraints imposed by the tuning parameters.

In addition to inhibiting over-adaptation to the observations in the training data, tuning may be suitable in cross-study prediction settings for inhibiting an over-adaptation of the model to the distribution of the training data, not only to the specific observations in the training data. The value of a tuning parameter optimal for cross-study prediction can be expected to lead to a simpler prediction rule. This is because a prediction rule optimal in the context of cross-study prediction should generalize well to independent datasets, which tend to share only coarse properties with the training data.

In the following an illustration is given of the discrepancy between conventional tuning and tuning for cross-study prediction by means of a fictional example. Prediction rule A obtained using the NSC algorithm on a specific dataset features optimal generalizability to observations from the same source when the shrinkage parameter Δ takes the value 1. When performing shrinkage using the tuning parameter value of 1, 100 genes are influential in the resulting prediction rule A. The prediction rule B obtained when using a large shrinkage parameter value, namely $\Delta = 3$, features optimal generalizability to sources outside of the training data in the mean. For prediction rule B the shrinkage leads to only 10 influential genes. Although this

number of genes is smaller, the influence of the genes is robust in the sense that it is so strong that it exists in any data from studies of the biological phenomenon of interest—despite batch effects.

Conventionally, tuning parameters are optimized using CV, which is suitable when the test data follow the same distribution as the training data but not in the case of cross-study prediction settings. While it certainly is desirable to choose proper values of the tuning parameters for cross-study prediction scenarios, it is unclear how this can be performed. If in addition to the training dataset there are several other datasets available featuring the same covariates and target variable, the respective tuning parameter could be optimized simply by minimizing the prediction error rate obtained for the other datasets. While this procedure certainly would be optimal, it is hardly ever practicable for a prediction problem at hand, because in practice the required independent datasets are almost never available.

In many situations there is one independent test dataset from a different source available. When such an independent dataset is at hand, the following would be possible: All parameters of the corresponding model are estimated using the training data with the exception of the tuning parameter, for which, using grid search, the value that minimizes the prediction error rate obtained for the independent test dataset is chosen. The goal of this procedure is to obtain a prediction rule that is general enough to apply well to sources different from that of the training data.

Note that when the independent test dataset is used both for tuning and for estimating the prediction error rate, the resulting error rate estimate underestimates the error rate to be expected in the case of new observations. The reason for the overoptimism of this estimate is that the value of the optimized tuning parameter is overly strongly adapted to the independent test dataset used for error estimation. Nevertheless, the degree of this overoptimism might be small enough to be negligible in practice. Otherwise, such overoptimism could be prevented by considering a second independent dataset and using this only to estimate the prediction error of the prediction rule, optimizing the tuning parameter using the first independent dataset. However, in practical situations it is probably rare that two independent datasets are available. If it becomes apparent that the approach of tuning using external data described above works considerably better than the conventional approach, the following would be possible to obtain a conservative error estimate: Perform conventional tuning using the training dataset, apply the resulting prediction rule to the test dataset and use the error rate obtained there as an error estimate. Note that although conventional tuning would be performed for error estimation here, the final prediction rule nevertheless would be obtained using tuning with external data.

A drawback of the approach of using a different dataset for tuning might be that

the tuning parameter value optimized using the independent dataset might not be appropriate for other independent datasets from the application field under consideration. Extensive studies would be necessary to refute that this risk is substantial.

Another way to obtain tuning parameter values suitable for cross-study prediction could be to provide empirically motivated rules of thumb for the choice of these parameters, for example "use two times the optimized value of Δ in NSC if weak batch effects are expected and three times this value if strong batch effects are expected".

In addition to the examples mentioned in section 4.4.3, another interesting tuning parameter could be the terminal node size of the classification trees constituting a RF, where larger terminal node sizes would correspond to a weaker adaptation of the classifier to the training data distribution. RFs generally are known to be able to capture complex dependence patterns present in the data; consequently, they could be prone to overfitting the training data distribution. Basically, the individual classification trees constituting a RF feature a high level of variability, but a small bias with respect to rendering the actual dependence structure of the target variable on the covariates. The high variability inherent in the individual classification trees is remedied in the final classifications by aggregating the individual trees, where the advantage of the small bias of the individual trees with respect to the true dependence structure is still retained. The ability of RFs potentially to reveal important aspects of the actual dependence structure underlying specific datasets makes them very appealing for complex modern biomolecular data. The working group "Computational Molecular Medicine" at the Department of Medical Informatics, Biometry and Epidemiology, University of Munich, to which I belong to, has conducted and will continue to conduct research on extensions of the RF methodology.

In the FAbatch estimation algorithm, the shrinkage parameter of the L_2-penalized logistic regression models was optimized using CV. The cross-batch and cross-study prediction performance of FAbatch might be improved by a tuning procedure of the shrinkage parameter that takes into account that the estimations of the class probabilities are made across batches or datasets. More precisely, the following procedure might be considered: using grid search choose the tuning parameter value that delivers the smallest error of the CV-related procedure used to estimate the class probabilities. See again the second step of the FAbatch estimation algorithm (section 3.2.2) for a description of this CV-related procedure. However, this procedure requires more than one batch to be present in the training data.

The idea of using external data in the construction of prediction rules has been realized by Xiao et al. (2014), who introduce a modification of RF in which the individual trees are weighted based on their performance of predicting the external data. This approach makes even more use of the test data than the approach of tuning

using external data described above, because the influence of variables relevant in the training data only and not in the test data is directly reduced and the influence of variables relevant in both training data and test data increases. By contrast, the aim of prediction with tuning using external data is to retain only variables whose influence is strong enough to be both relevant in training data and test data. Here it is assumed that variables with strong influences also are relevant to datasets other than the training data, while weaker influences correspond to artifacts specific to the training data. However, there also may be variables which, due to confounding with clinical covariates, exhibit a strong apparent relationship to the target variable in the training data that cannot be generalized to other datasets. Nevertheless, since it uses more information from the external dataset, the approach by Xiao et al. (2014) might result in a prediction rule that is overly adjusted to the specific external dataset used to be valuable in cross-study prediction. Variables relevant in the training data, but not in the specific test dataset at hand, might be relevant for other external data. Conversely, variables that are relevant in the test data at hand might be less influential for other external data. Prediction with tuning using external data, by contrast, requires less information from the specific external dataset used and thus might generalize better to other external datasets.

In the empirical study on cross-study prediction, all 25 publicly available datasets used had been collected using the traditional microarray data chip type HG-U133PLUS2. As previously stated, batch effects are stronger when using such traditional microarrays than when using more modern biomolecular data types. It cannot be precluded that addon batch effect adjustment is not beneficial or even slightly harmful when considering prediction rules based on other data types. Batch effect adjustment is, by necessity, connected with some variability, because the parameters involved are unknown and have to be estimated. In the case of very weak batch effects, there may not be much benefit from addon batch effect adjustment. Here, the variability connected with estimating the parameters involved in batch effect adjustment might lead to a small deterioration in prediction performance. With the aid of a real data analysis, whether this problem exists in actual applications to modern biomolecular data types can be studied. For this purpose, one could use the same analysis design as that presented in Chapter 4, replacing the datasets used there by datasets of a more modern type, for example by RNA-Seq datasets. Note that in section 3.3.2 it was reported that addon batch effect adjustment had been successfully applied to a dataset of a more modern data type in cross-batch prediction, namely miRNA measurements, using the Illumina Human-6 v2 Expression BeadChip. Nevertheless, the analysis outlined in section 3.3.2 can be regarded as illustrative only, since only a single dataset was considered there.

Just as batch effect adjustment is connected with variability, so is normalization; this is because in normalization the parameters involved have to be estimated from the data as well. Frozen RMA (McCall et al.; 2010) is an extension of RMA in which for normalizing the data at hand, the parameters involved are estimated using large amounts of data from publicly available databases. Frozen RMA performs addon quantile normalization, where the training of the quantile normalization is performed on the publicly available data. Frozen RMA has the advantage over RMA with addon normalization that no parameters have to be estimated using the data investigated, which is why this procedure is more robust. On the other hand, RMA with addon normalization has the advantage of being sensitive to the specific behavior of the training dataset.

The performance of prediction rules obtained when using frozen RMA as the normalization method has been compared to that of prediction rules obtained when using separate RMA normalization. Frozen RMA achieved slightly better results (see Parker and Leek (2012) for details); however, the batch effects in the two datasets considered in the study by Parker and Leek were only mild. It still might be interesting to compare frozen RMA and RMA with addon quantile normalization with respect to their performance in cross-study prediction in an extensive real data study comparable to that outlined in Chapter 4. Nevertheless large differences are not expected between these two approaches in case of the practically more interesting setting with large training datasets because here the result of RMA on the training datasets should be quite stable.

In all three main projects conducted in the context of this thesis, large collections of real datasets were used to evaluate the behavior of statistical methods in actual applications. For two analyses, in addition to studying real datasets, simulations were performed, namely for the large comparison study of the batch effect adjustment methods (sections 3.2.4 and 3.3.1) and for the comparison study of these methods in the context of cross-batch prediction (section 3.3.2). Although in most cases conclusions drawn from the simulation also were observable in the real data analysis, there sometimes were critical differences, for example with respect to the performance of standardization in cross-batch prediction. This illustrates that, as discussed in Boulesteix et al. (2015), it can be dangerous to rely completely on simulations. It is likely that in many papers in which empirical results based on simulations only are presented, false conclusions are drawn. Therefore, it seems like a valid strategy to study real datasets as well as simulated datasets to avoid drawing misleading conclusions from simulated data. Note, however, that in order to base conclusions on real data alone, a sufficiently large number of datasets are needed (see e.g. Boulesteix et al. (2015)).

A. Appendices to Chapter 2

A.1. Simulation study for the example of supervised variable selection

As described in section 2.4.3, a simulation study was conducted to investigate basic statistical properties of $\text{CVIIM}_{\boldsymbol{s},n,K}$.

Simulation design

To make the simulation design representative of high-dimensional biomedical data, the transcriptomic dataset `ProstatecTranscr` was employed to estimate realistic parameters to be used in the data-generating process. Datasets of sizes $n = 50$ and $n = 100$ were generated with $p = 2000$ continuous variables and a binary target variable with balanced class frequencies.

Two settings were considered for the mean and covariance structure $MeanCov$ for the classes:

1. Scenario with strong signal:

   2000 correlated variables were generated, 200 of which were informative, and had class-specific means and covariances. First, 2000 of the 12625 variables of `ProstatecTranscr` were selected, namely those yielding the smallest p-values from two-sample t-tests between the observations from the two classes. From these again the 200 variables corresponding to the smallest p-values were selected: they were taken as the informative variables, and the remaining 1800 as the non-informative. For each informative variable the difference was calculated between the mean of the observations belonging to class 2 and that of the observations belonging to class 1, resulting in the vector $\hat{\boldsymbol{\delta}}$ of length 200. Furthermore the empirical covariance matrix of the informative variables was calculated separately for classes 1 and 2, denoted as $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{class1}}$ and as $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{class2}}$, respectively. In the simulation, the vector of the informative variables was drawn from $N(\mathbf{0}_{200}, \hat{\boldsymbol{\Sigma}}_{\boldsymbol{class1}})$ for observations from class 1 and from $N(\hat{\boldsymbol{\delta}}, \hat{\boldsymbol{\Sigma}}_{\boldsymbol{class2}})$ for

observations from class 2. Theoretically it would be possible to draw the values of all 1800 non-informative variables at once from a multivariate normal distribution with a covariance matrix estimated from the data. However, this is computationally intractable. Therefore, the 1800 non-informative variables were split into blocks of 200 variables, and for each of these blocks the empirical covariance matrix $\hat{\boldsymbol{\Sigma}}_{\mathbf{0}\boldsymbol{j}}$, $j = 1, \ldots, 9$, was calculated. In the simulation a vector from $N(\mathbf{0}_{200}, \hat{\boldsymbol{\Sigma}}_{\mathbf{0}\boldsymbol{j}})$ was drawn for $j = 1, \ldots, 9$ and subsequently these vectors were combined. Thus, the covariance matrix of the non-informative variables is block-diagonal with a block size of 200.

2. Scenario with weak signal:

   This scenario is conceptually equivalent to the previous one but is different in the following ways: Only 100 variables are informative, the entries in the mean vector $\hat{\boldsymbol{\delta}}$ for class 2 from scenario 1, corresponding to the 100 informative variables, were multiplied by 0.7 and the block sizes for the non-informative variables were reduced from 200 to 100.

For the supervised variable selection two numbers of selected variables *psel* were considered: $psel = 10$ and $psel = 1000$. The variables yielding the smallest p-values from two-sample t-tests between the observations from the two classes were selected. Linear discriminant analysis was the classification method employed with $psel = 10$ variables and diagonal linear discriminant analysis was the classification method employed with $psel = 1000$. Again, the following commonly used splitting ratios between the sizes of the training sets and test sets were considered: 2:1 (3-fold CV), 4:1 (5-fold CV) and 9:1 (10-fold CV). Again, K in the following denotes the number of folds in the CV.

The simulation was performed for each possible combination of $MeanCov$, n, *psel* and K, leading to 24 simulation settings in total. For each setting 2000 datasets were simulated and for each the estimate $\text{CVIIM}_{\boldsymbol{s},n,K}$ was calculated. As with the real-data analyses presented in Chapter 2 the full and incomplete CV was repeated 300 times for each simulated dataset.

For approximating the true measure $\text{CVIIM}_{P,n,K}$ both $\mathbb{E}[e_{full,K}(\boldsymbol{S})]$ and $\mathbb{E}[e_{incompl,K}(\boldsymbol{S})]$ were approximated based on 10^5 simulated datasets of size n. Each dataset was randomly split into a training set of size $n_{train} := \lceil n(K-1)/K \rceil$ and a test set of size $n - n_{train}$. In the b-th iteration ($b = 1, \ldots, 10^5$) the approximation was done as follows: For $\mathbb{E}[e_{full,K}(\boldsymbol{S})]$ the variable selection and the training of the classifier were performed on the training set only and the resulting classifier was subsequently applied to the test set to calculate the error rate. The same procedure was followed for $\mathbb{E}[e_{incompl,K}(\boldsymbol{S})]$ with the exception that the variable selection was

performed on the entire dataset. By averaging over the 10^5 simulated datasets, close approximations of the true values of $\mathbb{E}[e_{full,K}(\boldsymbol{S})]$ and $\mathbb{E}[e_{incompl,K}(\boldsymbol{S})]$ are expected.

Results

Figure A.1 shows boxplots of the $\text{CVIIM}_{\boldsymbol{s},n,K}$ values for all simulation settings. The bias with respect to the true measure values $\text{CVIIM}_{P,n,K}$ is negligible in all settings. However, the variance around the true values is relatively large in many of the settings. Note that when computing CVIIM over multiple datasets, as one would in an extensive real-data analysis, the variability within the given distribution (as examined in this simulation study) and the variability across the datasets are measured.

The dependence of $\text{CVIIM}_{P,n,K}$ on the individual simulation parameters can be assessed better by examining Figure A.2. The number of observations n has a negative effect on CVIIM in all cases. An important and slightly surprising observation is that the results suggest no or only a slight dependence on the number of folds K. Higher values of CVIIM are observed when selecting 1000 variables, but this should not be over-interpreted since it may result from the specific simulation design. In the supervised variable selection analyses on real datasets performed in Chapter 2 this observation was not made. The influence of the mean-covariance structure $MeanCov$ depends on $psel$ (see Figure A.1). For $psel = 10$ smaller $\text{CVIIM}_{\boldsymbol{s},n,K}$ values are observed in the scenario with weak effects than in the scenario with strong effects; for $psel = 1000$ it is the reverse. This might be explained by the fact that in the scenario with weak effects there are only 100 informative variables. When selecting 1000 variables more noise variables are selected, impacting $\mathbb{E}[e_{full,K}(\boldsymbol{S})]$—causing the error to be larger—much more than $\mathbb{E}[e_{incompl,K}(\boldsymbol{S})]$.

The dependence of the variance of $\text{CVIIM}_{\boldsymbol{s},n,K}$ on the simulation parameters and on $\text{CVIIM}_{P,n,K}$ is visualized in Figure A.3. Unsurprisingly, the variance decreases with increasing n and increases with the number of folds K. The latter can be explained as follows: $\text{CVIIM}_{\boldsymbol{s},n,K}$ involves the fraction of two CV estimators which, with increasing K, become increasingly dataset-dependent—due to the training sets sharing more observations with the entire dataset—and therefore more variable. In the scenario with the stronger effects generally larger variances are observed. When selecting only $psel = 10$ variables, the variances are much higher than for $psel = 1000$.

For the scenario with $psel = 10$ a strong dependence of the variance on the true value of the measure can be observed, with smaller measure values leading to smaller variances. This dependence cannot be seen as clearly in the case of $psel = 1000$: A possible explanation is that the measure values are generally higher in this setting, obscuring the dependence at the relatively smaller CVIIM values.

The left panel of Figure A.4 suggests a dependence of $\text{CVIIM}_{P,n,K}$ on the true error $\mathbb{E}[e_{full,K}(\boldsymbol{S})]$, meaning that the use of the ratio of the errors in the calculation of CVIIM as opposed to their difference might not be sufficient to eliminate such an undesirable dependence. However this observed dependence may be explained by two observations regarding the simulation design. First, $\text{CVIIM}_{P,n,K}$ as well as $\mathbb{E}[e_{full,K}(\boldsymbol{S})]$ is larger for the smaller sample size $n = 50$, corresponding to the upper part of each ellipse. This negative dependence on n is natural (see section 2.4.5). Second, the upper-right ellipse, being responsible for much of the observed positive dependence of $\text{CVIIM}_{P,n,K}$ on $\mathbb{E}[e_{full,K}(\boldsymbol{S})]$, contains all scenarios with both weak effects and $psel = 1000$. As stated above, it can be suspected that the higher number of noisy variables selected in the case of $psel = 1000$ is responsible for the increase in $\mathbb{E}[e_{full,K}(\boldsymbol{S})]$ and $\text{CVIIM}_{P,n,K}$. The plot in the right panel of Figure A.4 suggests a much stronger dependence of $\max(\ \mathbb{E}[e_{full,K}(\boldsymbol{S})] - \mathbb{E}[e_{incompl,K}(\boldsymbol{S})],\ 0\ )$ on the true error. Here the values corresponding to the weak signal also are larger for $psel = 10$ and in the case of $psel = 1000$ the difference between the weak signals and strong signals is much bigger than for $\text{CVIIM}_{P,n,K}$.

In sections 2.2.7 and 2.4.2 an attempt is made to reflect the variance of $\text{CVIIM}_{\boldsymbol{s},n,K}$ through the use of the 25%- and 75%-quantiles of $\text{CVIIM}_{\boldsymbol{s},n,K,b} = 1 - e_{incompl,K}(\boldsymbol{s})_b / e_{full,K}(\boldsymbol{s})_b$, where index b indicates that these errors were obtained for run b (with $b = 1, \ldots, B$). Using the simulation results it is possible to investigate whether the variability in the $\text{CVIIM}_{\boldsymbol{s},n,K,b}$ values is indeed a meaningful surrogate for the variance of $\text{CVIIM}_{\boldsymbol{s},n,K}$. As a measure for the variability in the $\text{CVIIM}_{\boldsymbol{s},n,K,b}$ values, for each simulated dataset the empirical variance of the $\text{CVIIM}_{\boldsymbol{s},n,K,b}$ values ($b = 1, \ldots, 300$) can be calculated and defined as the *observed variability*. In Figure A.5 the values of the observed variability are plotted against the (approximated) true variance. Plots on the log-scale also are provided to enable comparisons of the small boxplots. In all plots the variance of the observed variability gets larger with a larger true variance. Moreover, the results clearly indicate that the size of the observed variability also is influenced strongly and positively by the size of the true variance. This dependence seems to be strongest for $K = 3$ and weakest for $K = 10$. This observed diminished relationship between observed variability and actual variance with increasing value of K becomes clearer when considering a fundamental shortcoming of the observed variability, which inhibits an even stronger relationship to the actual variance. The observed variability does not account for the fact that the error estimates in the B (incomplete) CV repetitions are dependent. For smaller training set sizes the individual CV estimates in $e_{full,K}(\boldsymbol{s})$ and in $e_{incompl,K}(\boldsymbol{s})$ are less similar, that is, less dependent. In these cases the observed variability thus better reflects the true variance. In contrast, in the case of larger training set sizes, the greater

dependence makes the behavior of the actual variance more different from that of the observed variability. These results suggest that the error bars obtained for the small K values are most appropriate for comparing the variability in individual $\text{CVIIM}_{\boldsymbol{s},n,K}$ values.

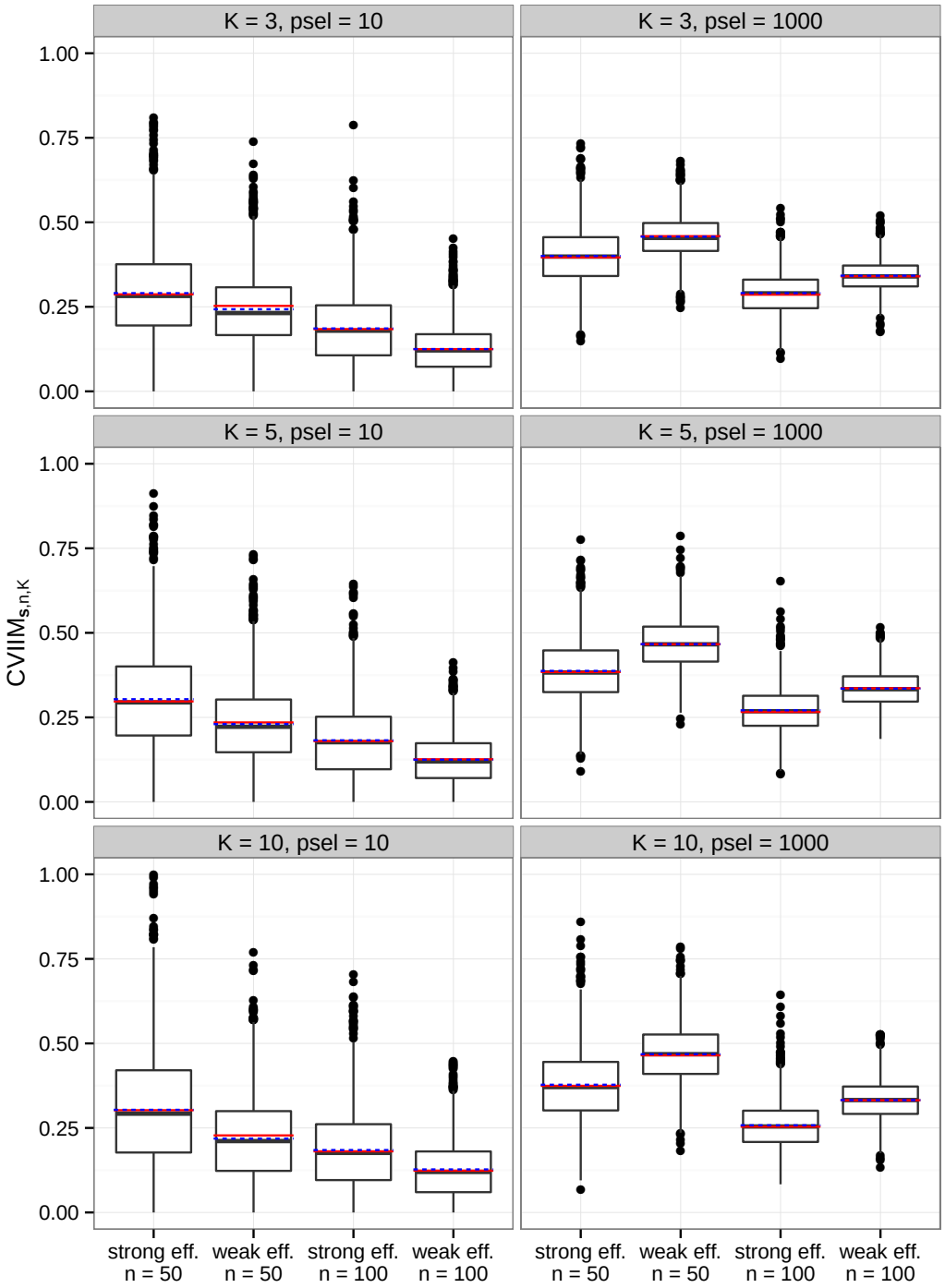

Figure A.1.: Estimated $\text{CVIIM}_{\boldsymbol{s},n,K}$ values from all simulation iterations (means depicted with broken blue lines) and true $\text{CVIIM}_{P,n,K}$ values (depicted with solid red lines)

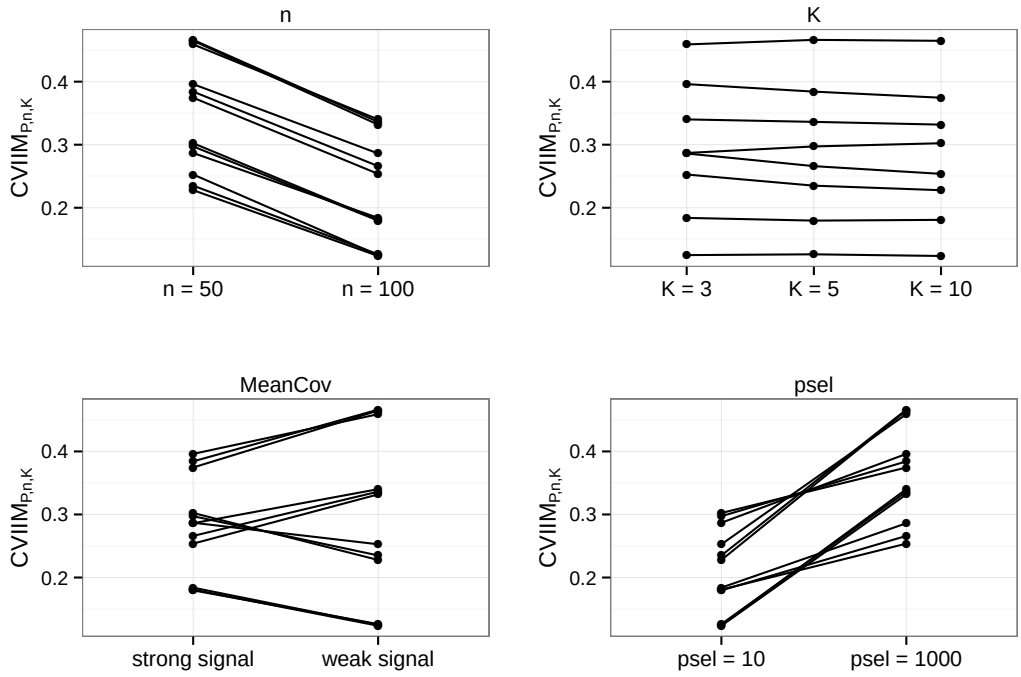

Figure A.2.: $\text{CVIIM}_{P,n,K}$ values for parameter settings, grouped according to n (top left), K (top right), $MeanCov$ (bottom left), and $psel$ (bottom right)

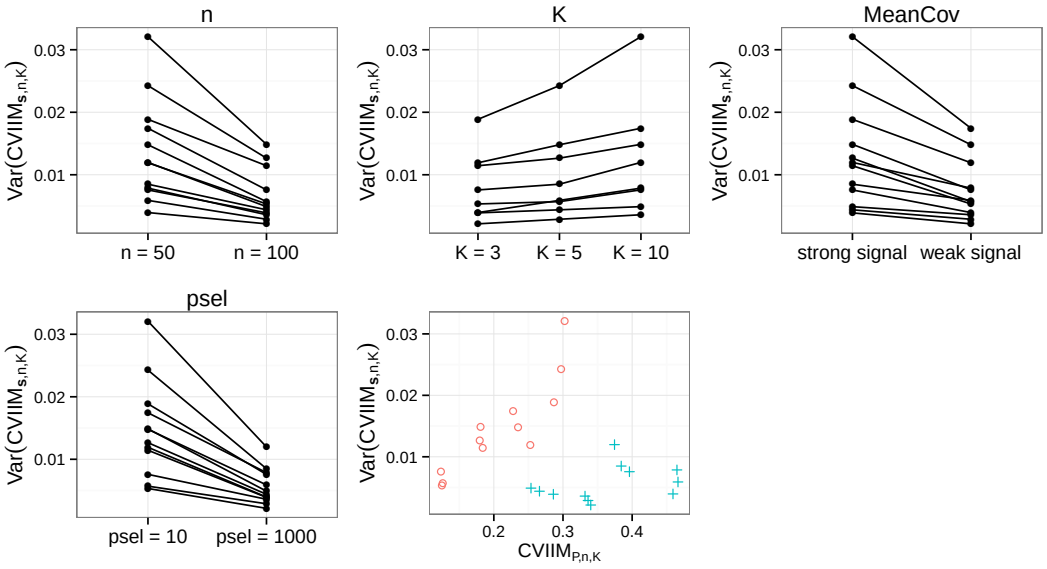

Figure A.3.: (Approximated) true variances of $\text{CVIIM}_{\boldsymbol{s},n,K}$ for parameter settings, grouped according to n, K, $MeanCov$, and $psel$, and scatterplot of the variance of $\text{CVIIM}_{\boldsymbol{s},n,K}$ versus the true $\text{CVIIM}_{P,n,K}$ values. The true variances are approximated by the empirical variances over the 2000 simulation iterations. The points corresponding to the setting with $psel = 10$ and $psel = 1000$ are depicted as red circles and cyan plus signs, respectively

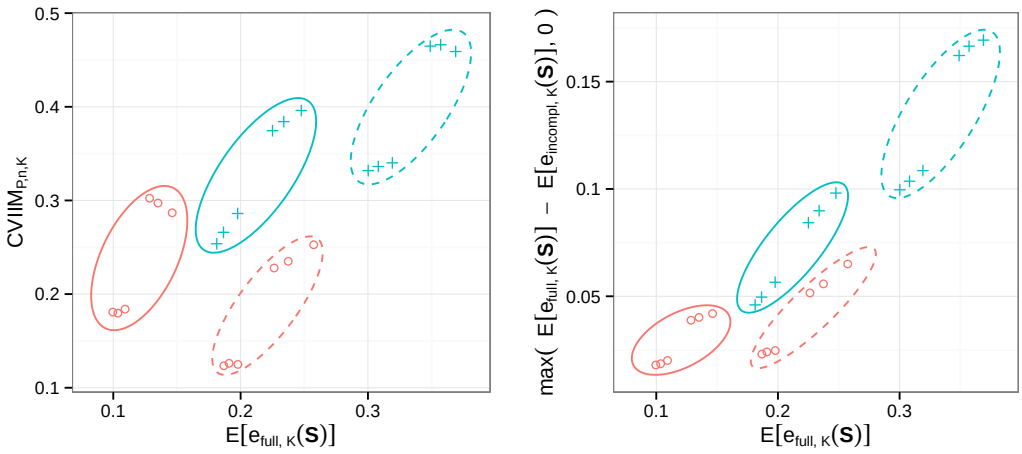

Figure A.4.: Values of $\text{CVIIM}_{P,n,K}$ (left) and $\mathbb{E}[e_{full,K}(\boldsymbol{S})]-\mathbb{E}[e_{incompl,K}(\boldsymbol{S})]$ (right) plotted against true errors. Each solid ellipse contains values corresponding to the strong signal and each dashed ellipse contains values corresponding to the weak signal. The points corresponding to the setting with $psel = 10$ and $psel = 1000$ are depicted as red circles and cyan plus signs, respectively

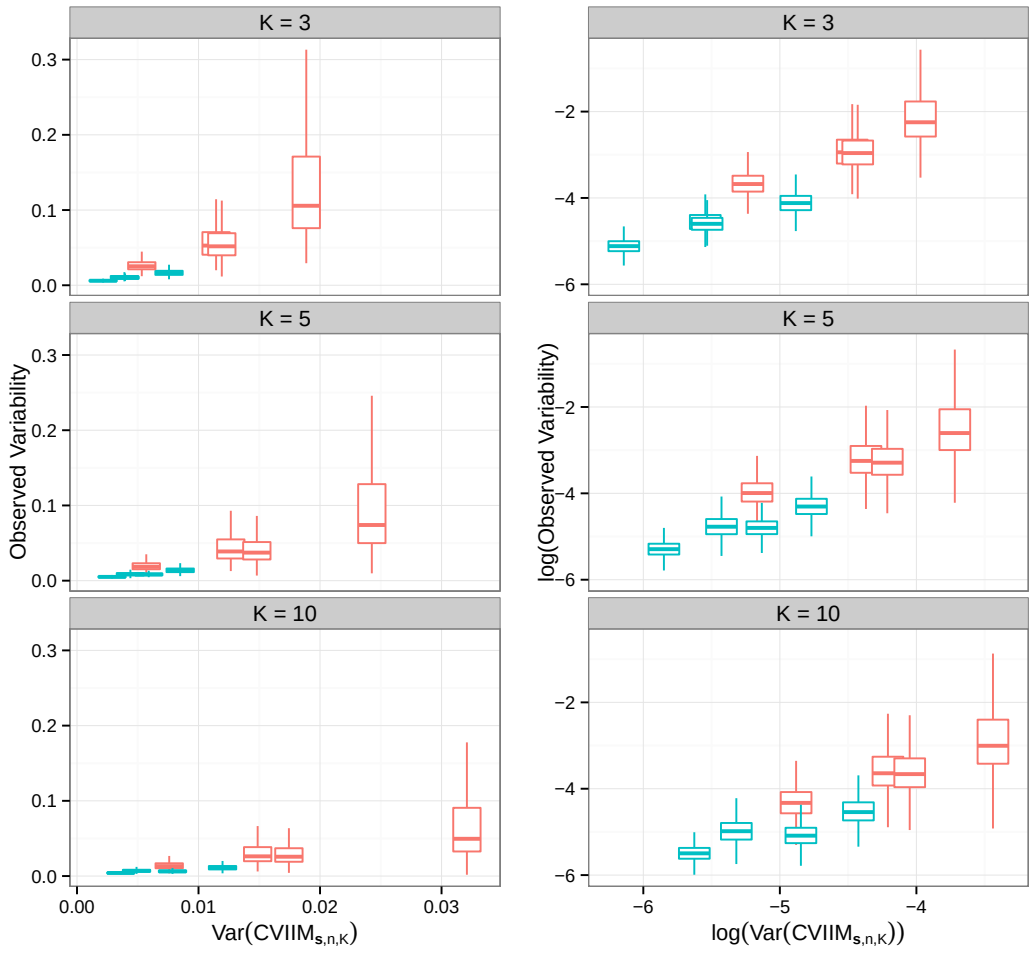

Figure A.5.: (Log-) values of the observed variability plotted against the actual (log-) variance of $\text{CVIIM}_{\boldsymbol{s},n,K}$ for different values of K. The boxplots corresponding to the setting with $psel = 10$ and $psel = 1000$ are red and cyan, respectively

A.2. Methodological background

A.2.1. Prediction rules, prediction errors, and their estimation

Let $\mathcal{X} \subset \mathbb{R}^p$ denote the predictor space and $\mathcal{Y} = \{1, 2\}$ the space of the response variable. Note that this notation also allows for categorical covariates, for example, $\mathcal{X} = \{0, 1\}^p \subset \mathbb{R}^p$ in the case of dummy-coded categorical predictors. Let $\boldsymbol{S} = \{(\boldsymbol{X}_1, Y_1), \ldots, (\boldsymbol{X}_n, Y_n)\}$ be an *i.i.d.* random sample with observations drawn from distribution P. Most importantly, here $\boldsymbol{x} \in \mathcal{X}$ denotes the "raw" data, meaning that these predictors may be subject to data preparation steps (possibly modifying their number or scale of measurement) before being used as predictors in classification.

A classification function $g : \mathcal{X} \mapsto \mathcal{Y}$, $\boldsymbol{x} \mapsto g(\boldsymbol{x})$ takes the vector $\boldsymbol{x}$ as an argument and returns a prediction $\hat{y}$ of the value y of the response variable. For example, in a classification task where, based on microarray samples, patients are classified as having cancer or not having cancer using the NSC approach, the corresponding function g would take the pre-normalized expression values as an argument, perform normalization, and classify the sample using a certain value of the shrinkage parameter. These steps are assumed to be performed in an ideal way, where all occurring parameters are estimated or optimized using a hypothetical dataset with sample size tending to infinity.

In practice g is estimated from the available data. Therefore $\hat{g}_{\boldsymbol{S}} : \mathcal{X} \mapsto \mathcal{Y}$, $\boldsymbol{x} \mapsto \hat{g}_{\boldsymbol{S}}(\boldsymbol{x})$ is defined as the classification function estimated from $\boldsymbol{S}$. In the example outlined above, this means that the parameters involved in the normalization procedure and the averages and variances involved in the nearest shrunken centroids classifier are estimated from $\boldsymbol{S}$ and that the shrinkage parameter also is chosen based on $\boldsymbol{S}$. The estimated classification function $\hat{g}_{\boldsymbol{S}}$ then can be used to predict y for a new observation.

Note that, as outlined in section 2.2.2, depending on the procedures involved in the estimation, it may not be straightforward to construct such a function $\hat{g}_{\boldsymbol{S}}$ that can be applied to predict independent data. However, from here on, it will be assumed that the necessary addon procedures (see section 2.2.2) are available and, thus, that the function $\hat{g}_{\boldsymbol{S}}$ can be constructed.

It is important to assess the prediction error of $\hat{g}_{\boldsymbol{S}}$, which is defined as

$$\varepsilon[\hat{g}_{\boldsymbol{S}}] := \mathbb{E}_{(\boldsymbol{X},Y)\sim P}[L(\hat{g}_{\boldsymbol{S}}(\boldsymbol{X}), Y)] = \int_{\mathcal{X}\times\mathcal{Y}} L(\hat{g}_{\boldsymbol{S}}(\boldsymbol{x}), y)\, dP(\boldsymbol{x}, y), \tag{A.1}$$

where $L(\cdot, \cdot)$ is an appropriate loss function, for example, the indicator loss yielding the misclassification error rate, as used in Chapter 2. The error defined in Eq. (A.1) is

commonly termed *conditional* because it refers to the specific sample $\boldsymbol{S}$. The *average error* across all samples following P^n is referred to as the unconditional error and denoted by $\varepsilon(n) \;:=\; \mathbb{E}_{\boldsymbol{S}\sim P^n}[\,\varepsilon[\hat{g}_{\boldsymbol{S}}]\,]$.

Let $\boldsymbol{s} = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$ denote a realization of the random sample $\boldsymbol{S}$. If a large independent sample were available, $\varepsilon[\hat{g}_{\boldsymbol{s}}]$ could be estimated directly by comparing the true values of the response variable in these data to the predictions made by $\hat{g}_{\boldsymbol{s}}$. Having only $\boldsymbol{s}$ available, a naïve approach would be to estimate $\varepsilon[\hat{g}_{\boldsymbol{s}}]$ using $\boldsymbol{s}$ itself as test data. This approach yields the *apparent error* or *resubstitution error*, which is known to be downwardly biased (i.e., too optimistic) as an estimator of $\varepsilon[\hat{g}_{\boldsymbol{s}}]$ since estimation of the classification function and error estimation are conducted on the same data. Resampling-based error estimation can be performed to address this issue. The sample $\boldsymbol{s}$ is split iteratively into non-overlapping training datasets and test datasets. In each iteration function g is estimated based on the training set, and the error of this estimated function is assessed based on the test set. In the following K-fold CV, the most widely used of these resampling-based approaches, is treated.

Given a random partition of the dataset $\boldsymbol{s}$ into K approximately equally sized folds $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_K$, the K-fold CV error estimate is given as

$$\frac{1}{K}\sum_{k=1}^{K}\frac{1}{\#\boldsymbol{s}_k}\sum_{j\,\in\,\{i\,:\,(\boldsymbol{x}_i,y_i)\,\in\,\boldsymbol{s}_k\}} L(\hat{g}_{\boldsymbol{s}\setminus\boldsymbol{s}_k}(\boldsymbol{x}_j), y_j), \tag{A.2}$$

where $\#$ represents the cardinality, $\boldsymbol{s}\setminus\boldsymbol{s}_k$ is the training set in iteration k and $\boldsymbol{s}_k$ is the test set. Since this estimate depends heavily on the considered random partition of the sample $\boldsymbol{s}$ into K folds, it is recommended to repeat this procedure $B > 1$ times and average the error estimates across the B repetitions. With $\boldsymbol{s}_{b1}, \ldots, \boldsymbol{s}_{bK}$ denoting the folds considered in the b-th repetition, the repeated K-fold CV error estimate is given as

$$e_K(\boldsymbol{s}) = \frac{1}{B}\sum_{b=1}^{B}\frac{1}{K}\sum_{k=1}^{K}\frac{1}{\#\boldsymbol{s}_{bk}}\sum_{j\,\in\,\{i\,:\,(\boldsymbol{x}_i,y_i)\,\in\,\boldsymbol{s}_{bk}\}} L(\hat{g}_{\boldsymbol{s}\setminus\boldsymbol{s}_{bk}}(\boldsymbol{x}_j), y_j). \tag{A.3}$$

If, for simplicity, it is assumed that the $\boldsymbol{s}_{b1}, \ldots, \boldsymbol{s}_{bK}$ $(b = 1, \ldots, B)$ are equally sized with size $n_{train,K} := \#\boldsymbol{s}\setminus\boldsymbol{s}_{bk}$ for $b \in \{1, \ldots, B\}$ and $k \in \{1, \ldots, K\}$, it easily can be seen that $e_K(\boldsymbol{s})$ is an unbiased estimator of $\varepsilon(n_{train,K})$ and therefore an upwardly biased estimator of $\varepsilon(n)$. This bias is called the *inherent bias* of CV in Varma and Simon (2006). Note that the notation $e_K(\boldsymbol{s})$ does not reflect the fact that the repeated K-fold CV error estimate depends on the random partitions in the B iterations. For purposes here, B is assumed to be large enough that this dependence can be ignored.

A.2.2. Incomplete opposed to full CV

With the issue of incomplete CV in mind, the following notation is introduced

$$\hat{g}_{\boldsymbol{a}_1}^{\boldsymbol{a}_2} : \mathcal{X} \mapsto \mathcal{Y} \qquad \boldsymbol{x} \mapsto \hat{g}_{\boldsymbol{a}_1}^{\boldsymbol{a}_2}(\boldsymbol{x}) \qquad \boldsymbol{a}_1 \subseteq \boldsymbol{a}_2 \subseteq \boldsymbol{s} \tag{A.4}$$

to denote an estimated classification function estimated based in part on a sample $\boldsymbol{a}_2$ and in part on a possibly smaller subsample $\boldsymbol{a}_1$ (i.e., one or several steps may be performed on a bigger sample). Returning to the example of microarray-based classification, it is common practice to run the normalization procedure, and often also the parameter tuning, based on the entire dataset $\boldsymbol{s}$ but to perform the training of the classifier within CV, that is, based only on the training set $\boldsymbol{s} \setminus \boldsymbol{s}_{bk}$ in each iteration k of each repetition b. In this scenario $\boldsymbol{a}_2$ would be the entire dataset $\boldsymbol{s}$ and in each CV iteration $\boldsymbol{a}_1$ would be the training set $\boldsymbol{s} \setminus \boldsymbol{s}_{bk}$.

With $\boldsymbol{a}_1 = \boldsymbol{s} \setminus \boldsymbol{s}_{bk}$ and $\boldsymbol{a}_2 = \boldsymbol{s}$ for $b = 1, \ldots, B$ and $k = 1, \ldots, K$, we obtain the incomplete CV error estimate, which is downwardly biased as an estimator of $\varepsilon(n_{train,K})$:

$$e_{incompl,K}(\boldsymbol{s}) := \frac{1}{B}\sum_{b=1}^{B}\frac{1}{K}\sum_{k=1}^{K}\frac{1}{\#\boldsymbol{s}_{bk}}\sum_{j \in \{i \,:\, (\boldsymbol{x}_i, y_i) \in \boldsymbol{s}_{bk}\}} L(\hat{g}^{\boldsymbol{s}}_{\boldsymbol{s}\setminus\boldsymbol{s}_{bk}}(\boldsymbol{x}_j), y_j), \tag{A.5}$$

where the index "incompl" indicates that the entire sample $\boldsymbol{s}$ is used for at least part of the data analysis steps required for the estimation of g, and that the resulting CV procedure is, thus, incomplete. The estimator $e_{incompl,K}(\boldsymbol{s})$ is unbiased as an estimator of the *average incomplete error* $\varepsilon_{incompl}(n_{train,K}; n) := \mathbb{E}_{\boldsymbol{S}\sim P^n}[L(\hat{g}^{\boldsymbol{S}}_{\boldsymbol{S}_{train,K}}(\boldsymbol{X}_{n_{train,K}+1}), Y_{n_{train,K}+1})]$, with $\boldsymbol{S}_{train,K} = \{(\boldsymbol{X}_1, Y_1), \ldots, (\boldsymbol{X}_{n_{train,K}}, Y_{n_{train,K}})\}$ and $(\boldsymbol{X}_{n_{train,K}+1}, Y_{n_{train,K}+1})$ playing the role of an arbitrary test set observation in $\boldsymbol{S}$. Here, exchangeability of the random observations in $\boldsymbol{S}$ is assumed.

Furthermore, since by definition $\hat{g}^{\boldsymbol{s}\setminus\boldsymbol{s}_{bk}}_{\boldsymbol{s}\setminus\boldsymbol{s}_{bk}} = \hat{g}_{\boldsymbol{s}\setminus\boldsymbol{s}_{bk}}$, the usual repeated K-fold error estimate from Eq. (A.3) is obtained if $\boldsymbol{a}_1 = \boldsymbol{a}_2 = \boldsymbol{s} \setminus \boldsymbol{s}_{bk}$ is set for $k = 1, \ldots, K$, and $b = 1, \ldots, B$. This estimator is denoted by $e_{full,K}(\boldsymbol{s})$:

$$e_{full,K}(\boldsymbol{s}) := e_K(\boldsymbol{s}) = \frac{1}{B}\sum_{b=1}^{B}\frac{1}{K}\sum_{k=1}^{K}\frac{1}{\#\boldsymbol{s}_{bk}}\sum_{j \in \{i \,:\, (\boldsymbol{x}_i, y_i) \in \boldsymbol{s}_{bk}\}} L(\hat{g}^{\boldsymbol{s}\setminus\boldsymbol{s}_{bk}}_{\boldsymbol{s}\setminus\boldsymbol{s}_{bk}}(\boldsymbol{x}_j), y_j), \tag{A.6}$$

where index "full" underlines that *all* steps of prediction rule construction are conducted within the CV procedure, that is, using the training sets only.

For easier interpretation, in Chapter 2 and in other sections of this Appendix

$\mathbb{E}[e_{full,K}(\boldsymbol{S})]$ is written for $\varepsilon(n_{train,K})$ and $\mathbb{E}[e_{incompl,K}(\boldsymbol{S})]$ for $\varepsilon_{incompl}(n_{train,K}; n)$.

A.2.3. Behavior of $\mathrm{CVIIM}_{\boldsymbol{s},n,K}$ for small $\varepsilon_{full}(n_{train,K})$ values

For very small values of $\varepsilon_{full}(n_{train,K})$, extreme CVIIM estimates can occur (either zero or very high values). For very small values of $e_{full,K}(\boldsymbol{s})$, the CVIIM estimate is highly sensitive to relatively small differences between $e_{incompl,K}(\boldsymbol{s})$ and $e_{full,K}(\boldsymbol{s})$, which may be due at least partly to random fluctuations. For example, suppose that $e_{full,K}(\boldsymbol{s}) = 0.01$ and $e_{incompl,K}(\boldsymbol{s}) = 0.001$, then there would be $\mathrm{CVIIM}_{\boldsymbol{s},n,K} = 0.9$. Note, however, that such extremely large results are expected to be rare due to a mechanism related to regression toward the mean: Considering the high level of variance of CV estimates, in many cases very small values of $e_{full,K}(\boldsymbol{s})$ are an underestimation of $\varepsilon_{full}(n_{train,K})$. In this case it is unlikely that $\varepsilon_{incompl}(n_{train,K}; n)$ is considerably more affected by underestimation. Thus, in such a situation it is unlikely that $e_{incompl,K}(\boldsymbol{s})$ is much smaller than $e_{full,K}(\boldsymbol{s})$. Instead, the incomplete CV error estimator $e_{incompl,K}(\boldsymbol{s})$ is more likely to be closer to its mean than $e_{full,K}(\boldsymbol{s})$, thereby preventing an overly large CVIIM estimate.

B. Appendices to Chapter 3

B.1. Plots used in verification of model assumptions

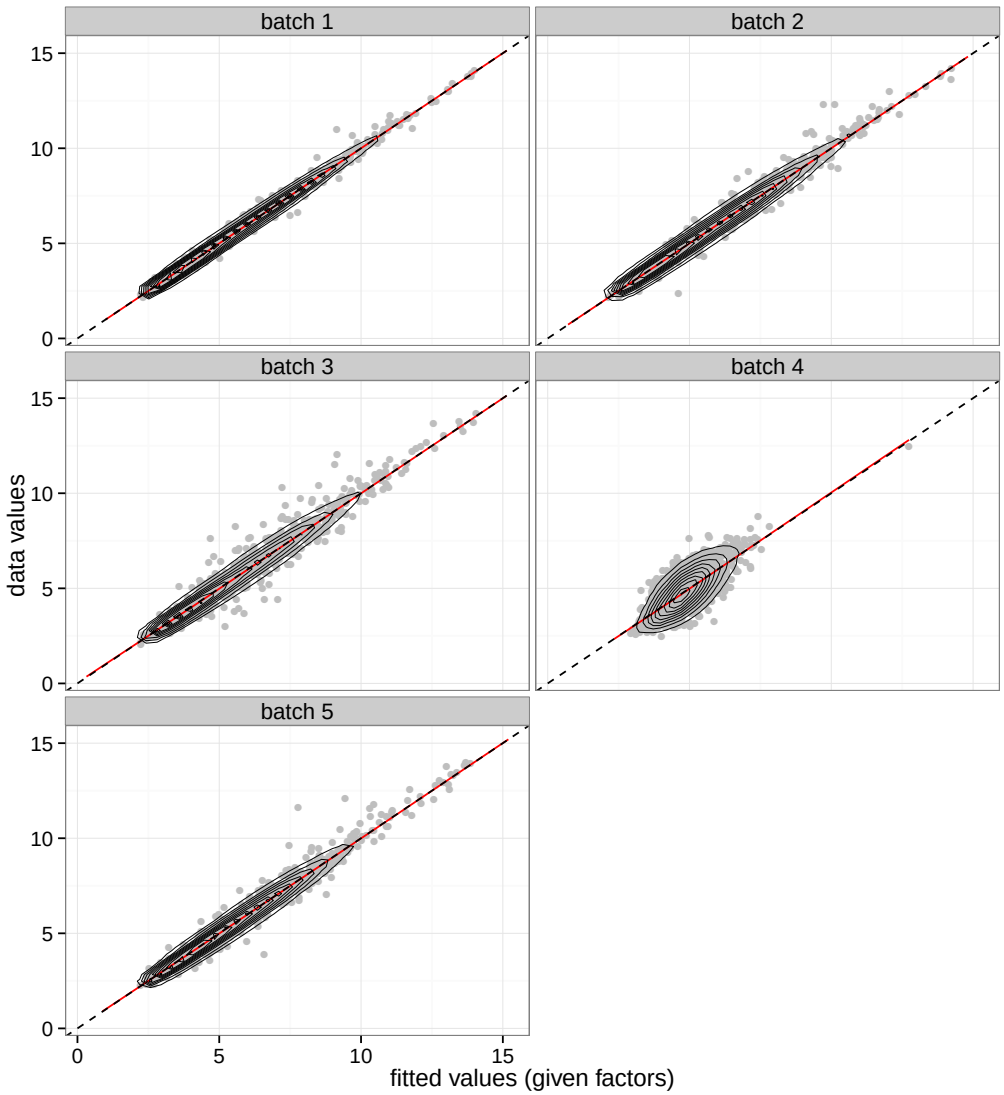

Figure B.1.: Data values against corresponding fitted values resulting from the FAbatch method. The contour lines represent two-dimensional kernel density estimates. The broken lines mark the bisectors and the red lines are LOESS estimates of the associations. The grey dots in each case are random subsets of size 1000 of all values.

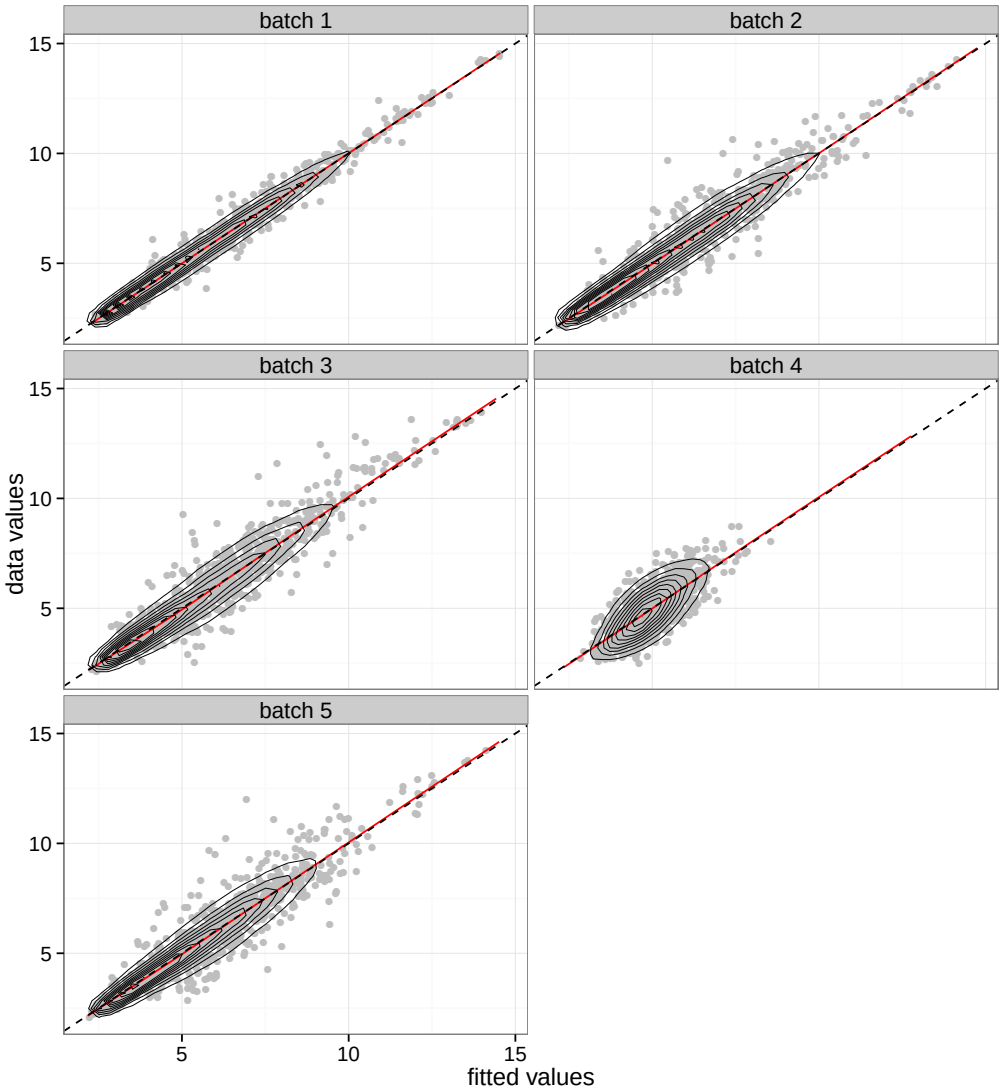

Figure B.2.: Data values against corresponding fitted values resulting from the ComBat method. The contour lines represent two-dimensional kernel density estimates. The broken lines mark the bisectors and the red lines are LOESS estimates of the associations. The grey dots in each case are random subsets of size 1000 of all values.

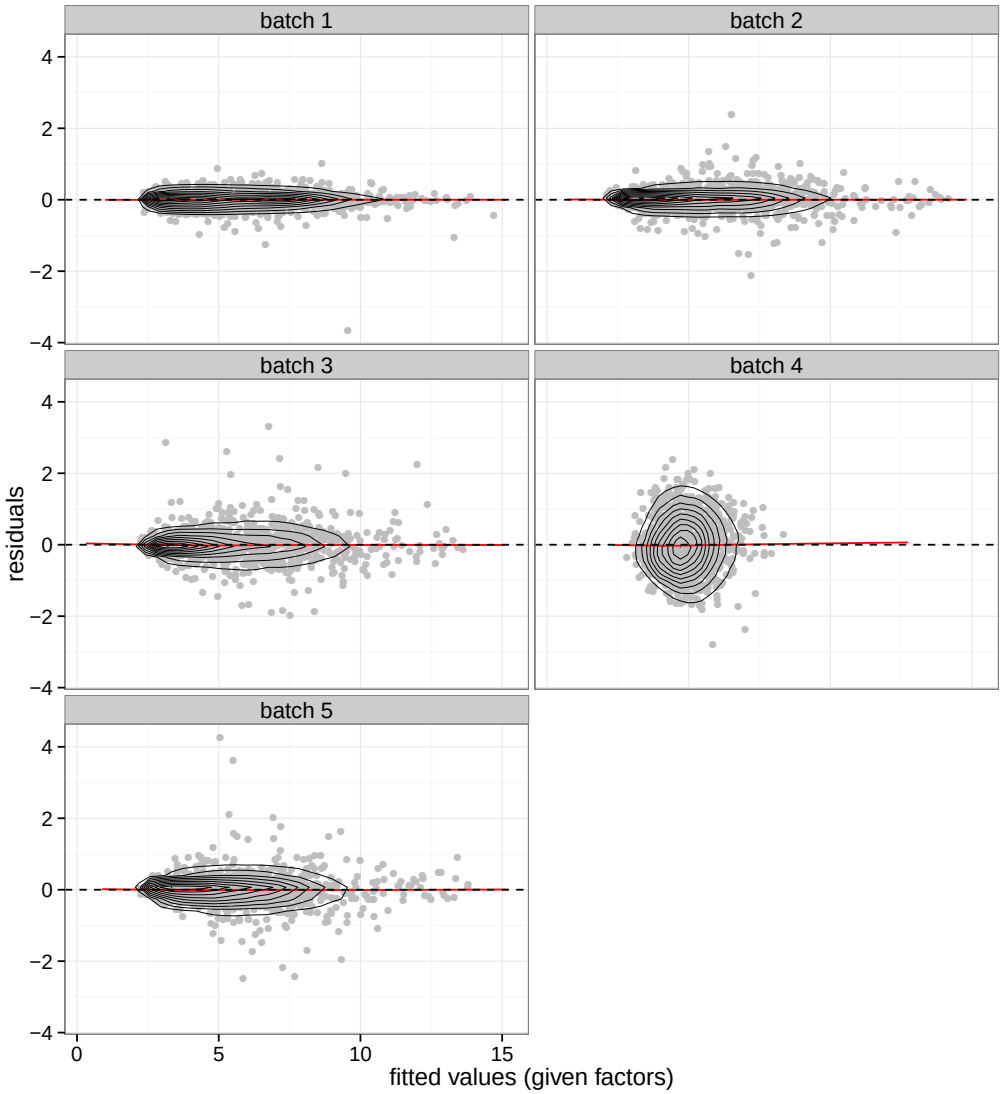

Figure B.3.: Deviations from fitted values resulting from the FAbatch method against corresponding fitted values. The contour lines represent two-dimensional kernel density estimates. The broken lines mark the horizontal zero lines and the red lines are LOESS estimates of the associations. The grey dots in each case are random subsets of size 1000 of all values.

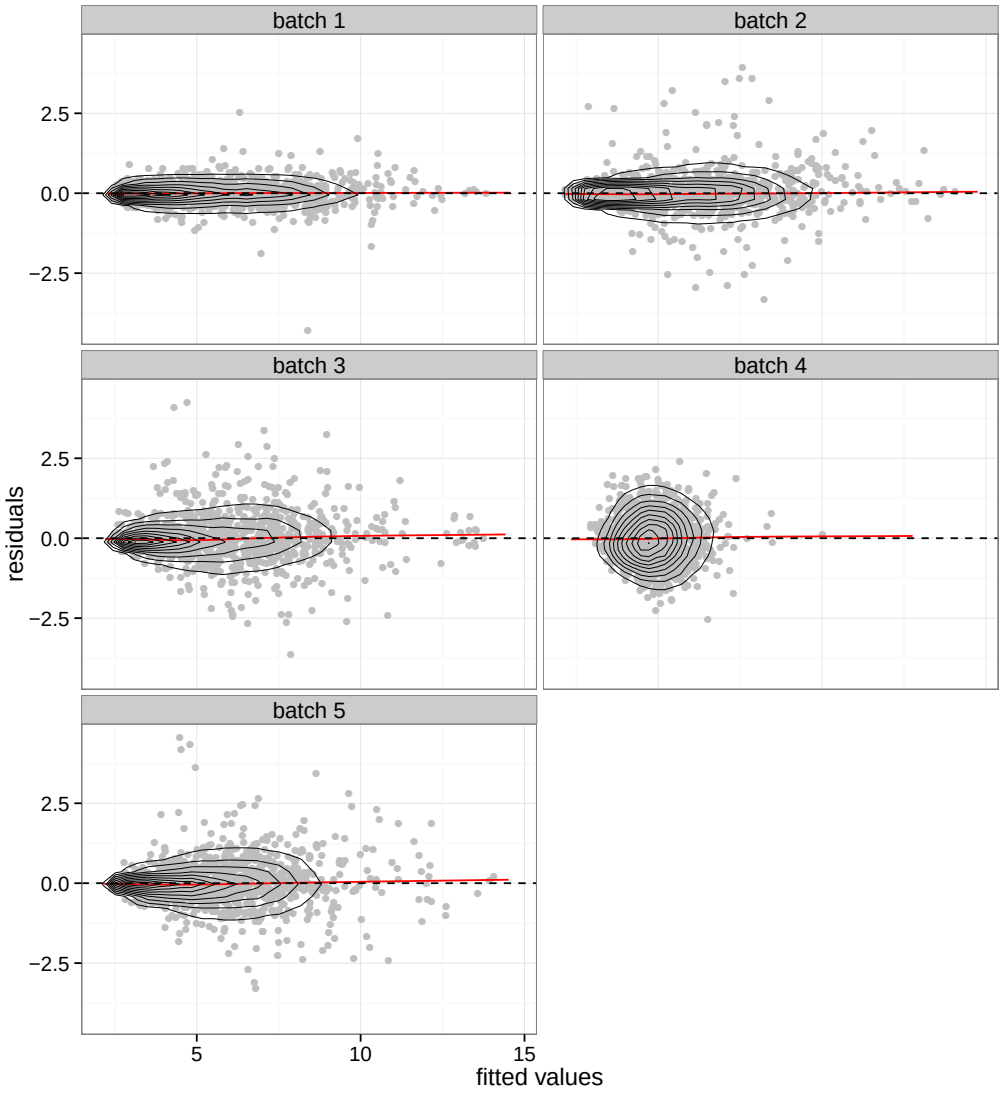

Figure B.4.: Deviations from fitted values resulting from the ComBat method against corresponding fitted values. The contour lines represent two-dimensional kernel density estimates. The broken lines mark the horizontal lines and the red lines are LOESS estimates of the associations. The grey dots in each case are random subsets of size 1000 of all values.

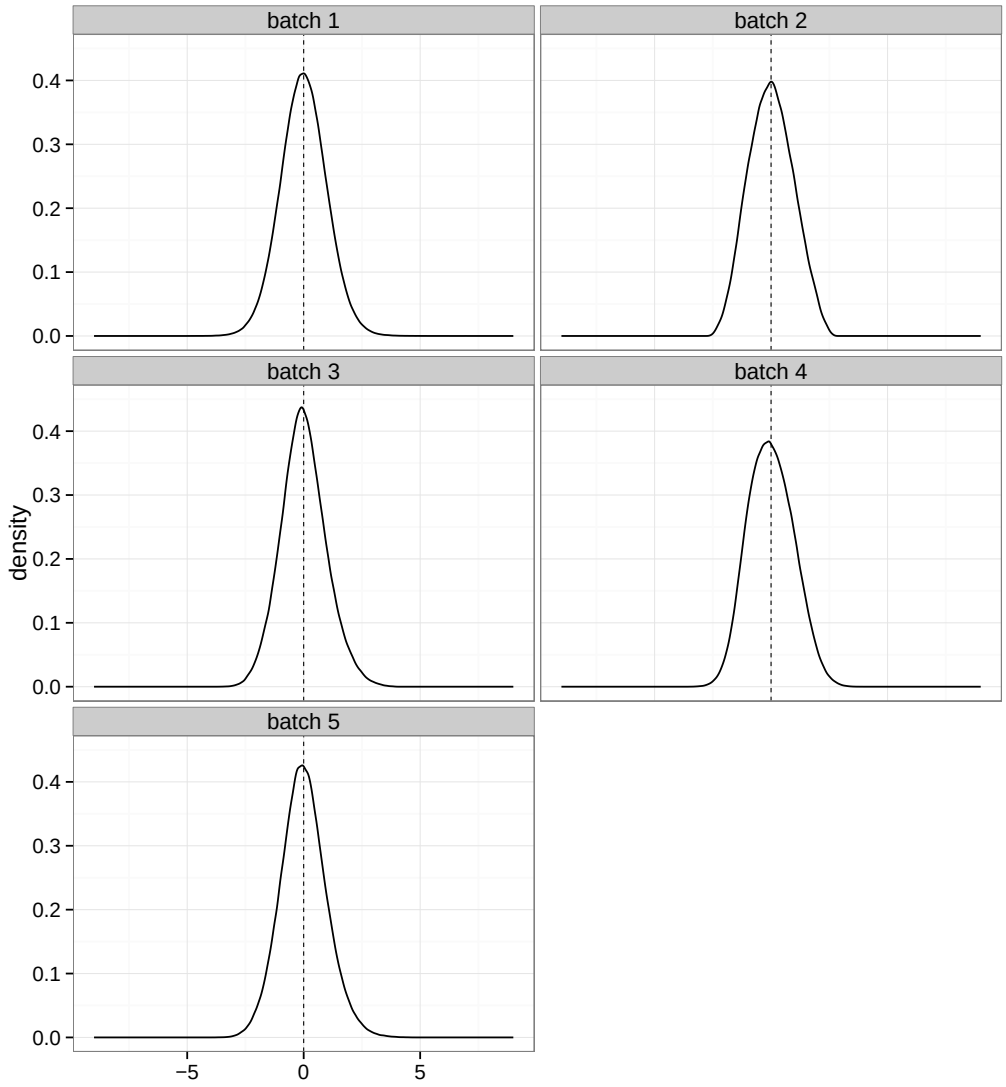

Figure B.5.: Density estimates of the deviations from the fitted values divided by their standard deviations for the FAbatch method. The broken lines mark the vertical zero lines.

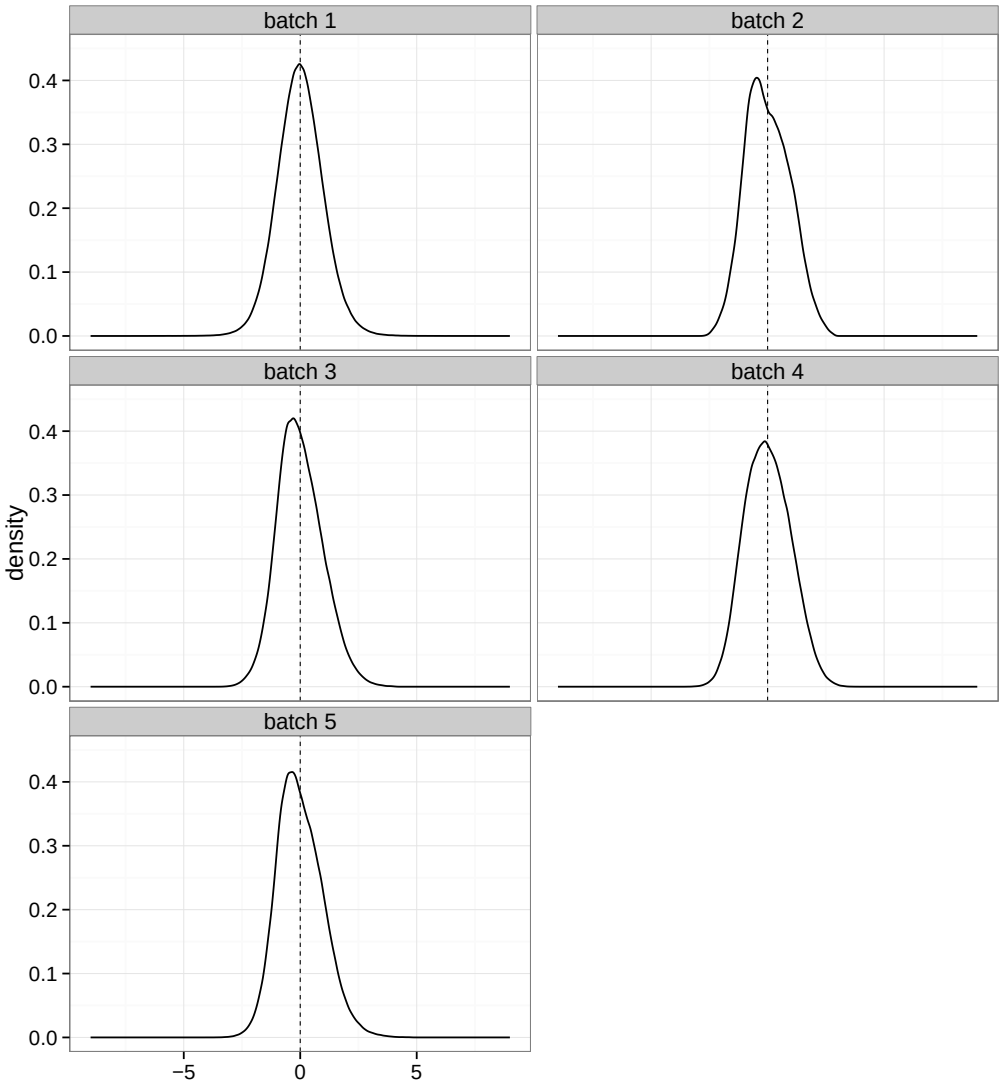

Figure B.6.: Density estimates of the deviations from the fitted values divided by their standard deviations for the ComBat method. The broken lines mark the vertical zero lines.

B.2. Target variables of datasets used in comparison study

ColonGastricEsophagealcSNPArray: "gastric cancer" ($y = 2$) versus "healthy" ($y = 1$)

AgeDichotomTranscr: "older than the median age of patients" ($y = 2$) versus "younger than or the same age as the median age of patients" ($y = 1$)

EthnicityMethyl: "Caucasian, from Utah and of European ancestry" ($y = 2$) versus "Yorubian, from Ibadan Nigeria" ($y = 1$)

BipolardisorderMethyl: "bipolar disorder" ($y = 2$) versus "healthy" ($y = 1$)

PostpartumDepressionMethyl: "depression post partum" ($y = 2$) versus "healthy" ($y = 1$)

AutismTranscr: "autistic" ($y = 2$) versus "healthy" ($y = 1$)

BreastcTranscr: "breast cancer" ($y = 2$) versus "healthy" ($y = 1$)

BreastCancerConcatenation: "breast cancer" ($y = 2$) versus "healthy" ($y = 1$)

IUGRTranscr: "intrauterine growth restriction" ($y = 2$) versus "healthy" ($y = 1$)

IBSTranscr: "constipation-predominant/diarrhoea-predominant irritable bowel syndrome" ($y = 2$) versus "healthy" ($y = 1$)

SarcoidosisTranscr: "sarcoidosis" ($y = 2$) versus "healthy" ($y = 1$)

pSSTranscr: "Sjogren's/sicca syndrome" ($y = 2$) versus "healthy" ($y = 1$)

AlcoholismTranscr: "chronic alcoholic" ($y = 2$) versus "healthy" ($y = 1$)

WestNileVirusTranscr: "severe West Nile virus infection" ($y = 2$) versus "asymptomatic West Nile virus infection" ($y = 1$)

B.3. Reasons for batch effect structures of datasets used in comparison study

EthnicityMethyl: "To limit the potential bias due to experimental batches, samples were randomized by population identity and hybridized in three batches." (Moen et al.; 2013)

BreastcTranscr: "To minimize possible processing and chip lot effects, samples were assigned to processing batches of seven to nine pairs, and batches had similar distributions of age, race, and date of enrollment. For array hybridization, each batch was assigned to one of two different chip lots ('A' and 'B') in a manner designed to ensure a balance of these same characteristics. [...] Laboratory personnel were blind to case control status and other phenotype information." (Godfrey et al.; 2013)

BreastCancerConcatenation: Concatenation of five independent datasets.

IUGRTranscr: Citation from the description on the ArrayExpress website: "[...] were collected during the years of 2004-2008 and hybridized in two batches to microarrays. Samples were randomized across arrays to control for array and batch variability." (ArrayExpress website)

AlcoholismTranscr: The batch variable in the sdrf.txt file is designated as "labeling batch", from which it was deduced that the batch structure is due to labeling for this dataset.

B.4. Boxplots of the metric values for simulated datasets for each method and simulation scenario

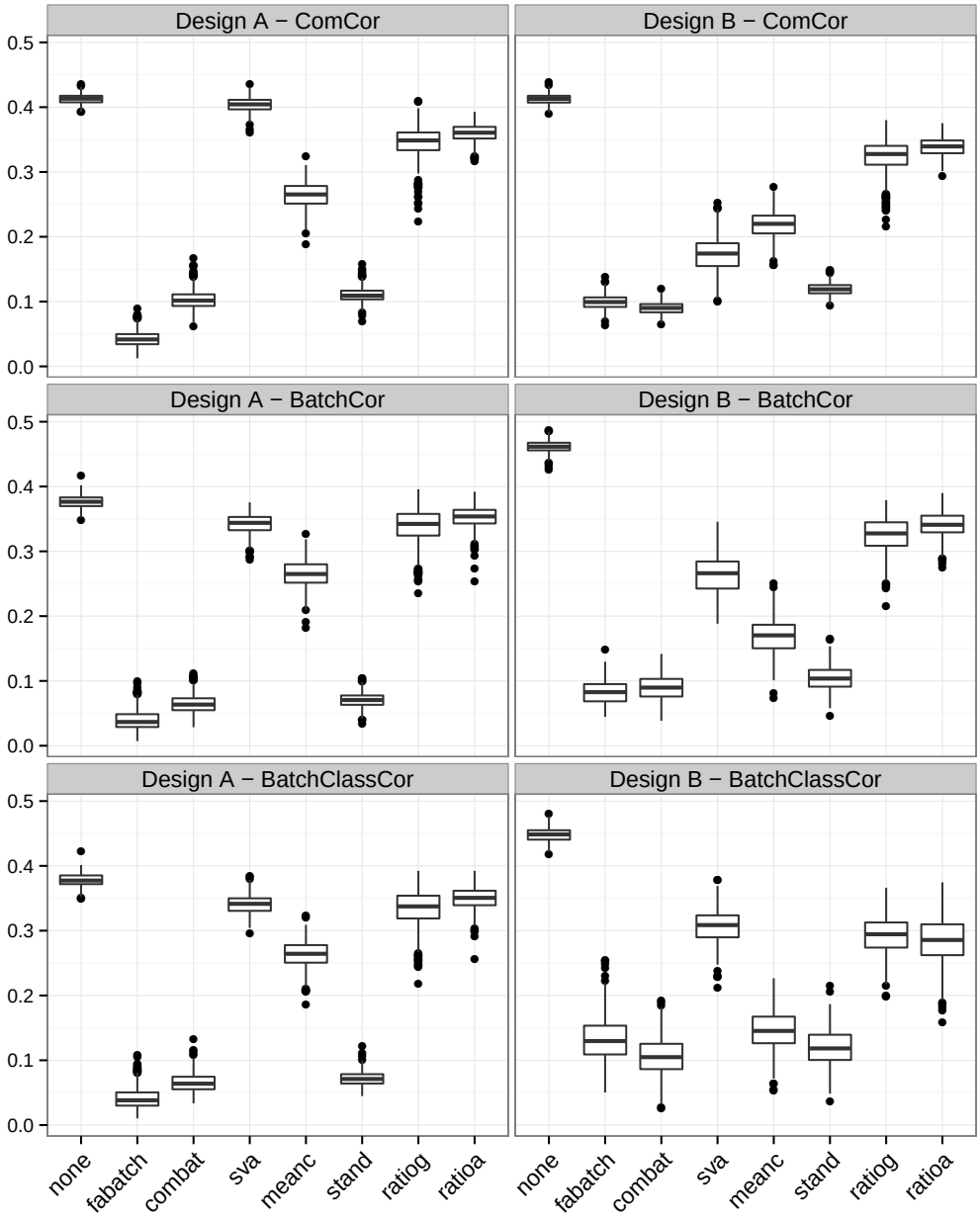

Figure B.7.: Values of metric `sepscore` for all simulated datasets separated by simulation scenario and by method.

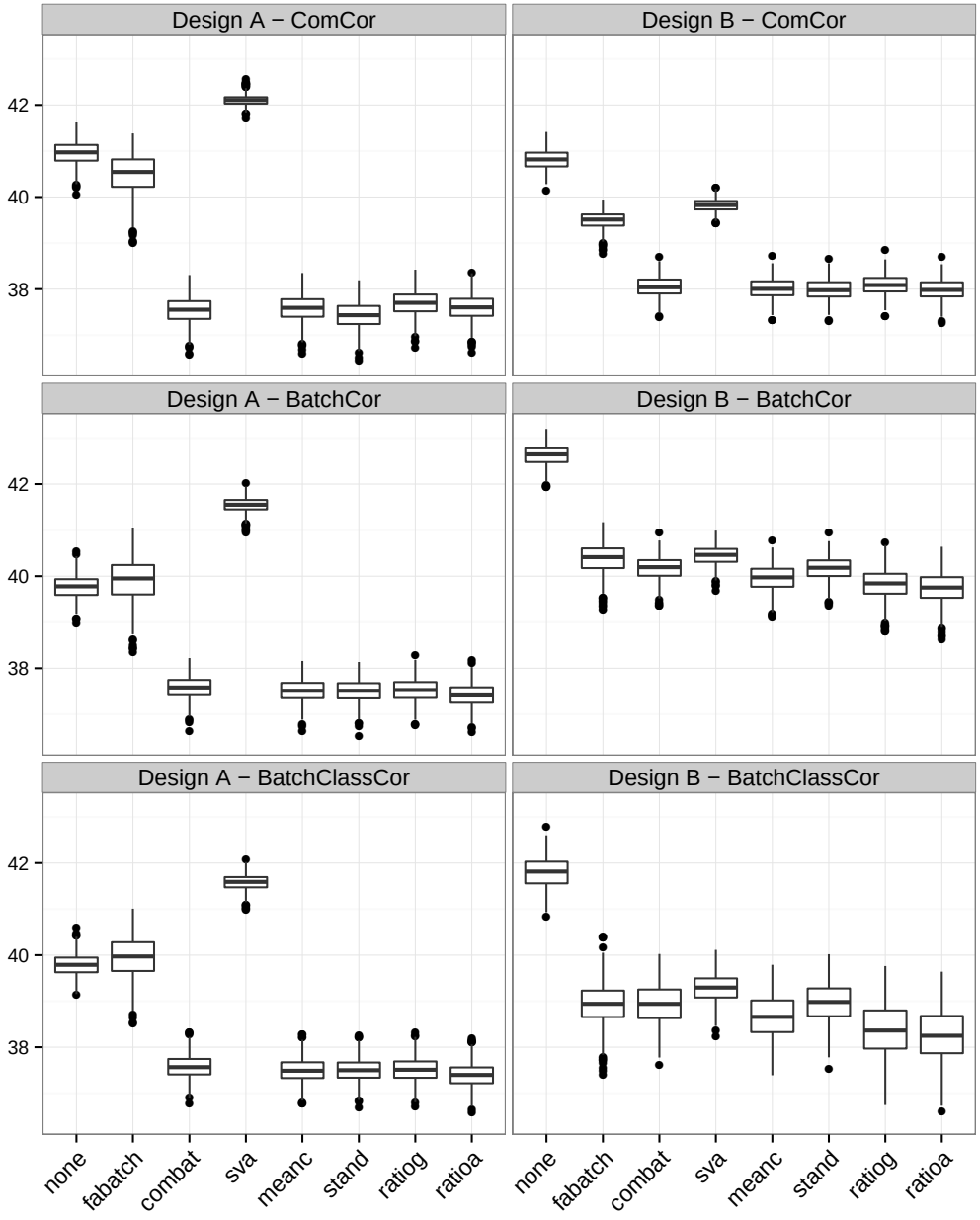

Figure B.8.: Values of metric `avedist` for all simulated datasets separated by simulation scenario and by method.

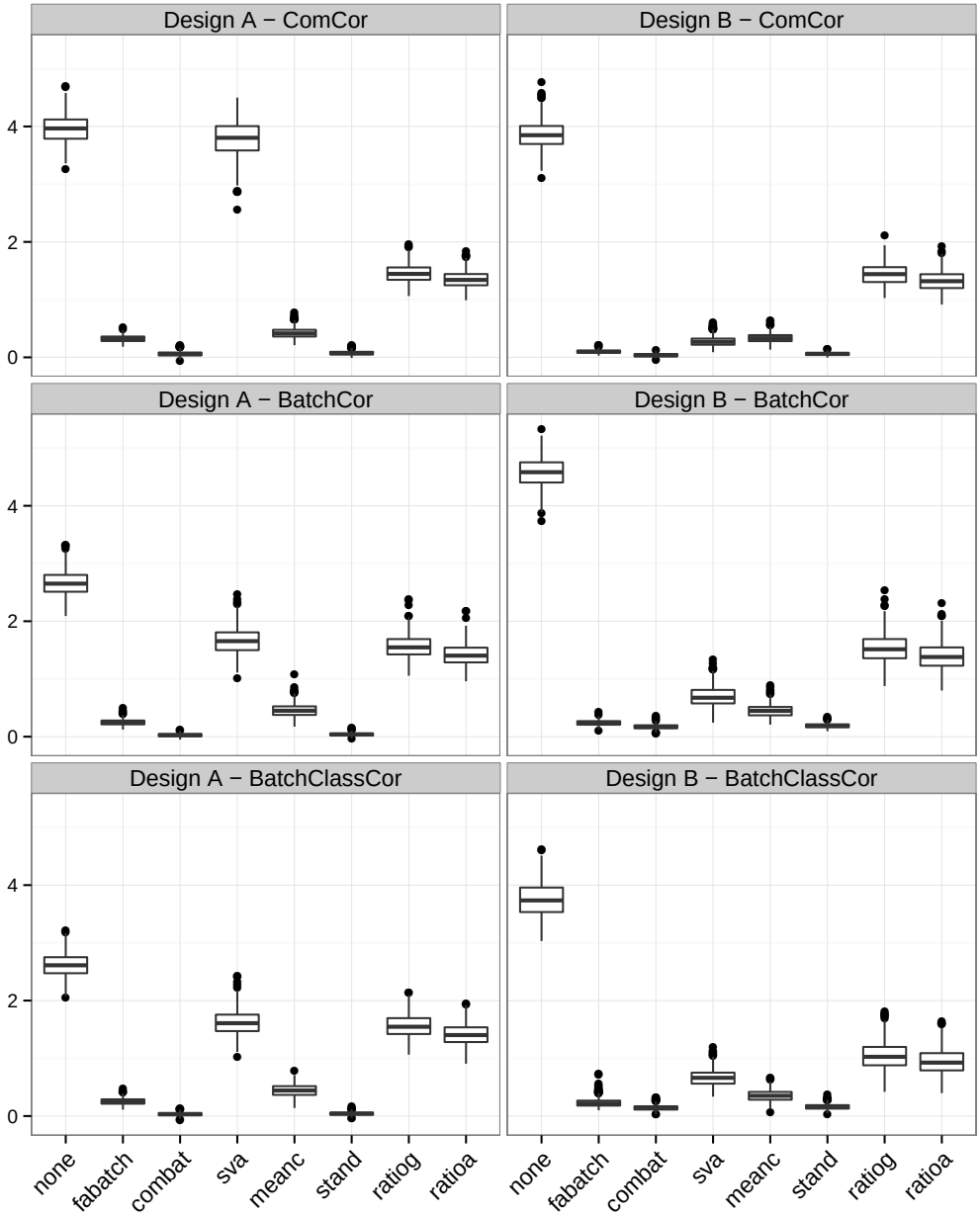

Figure B.9.: Values of metric `klmetr` for all simulated datasets separated by simulation scenario and by method.

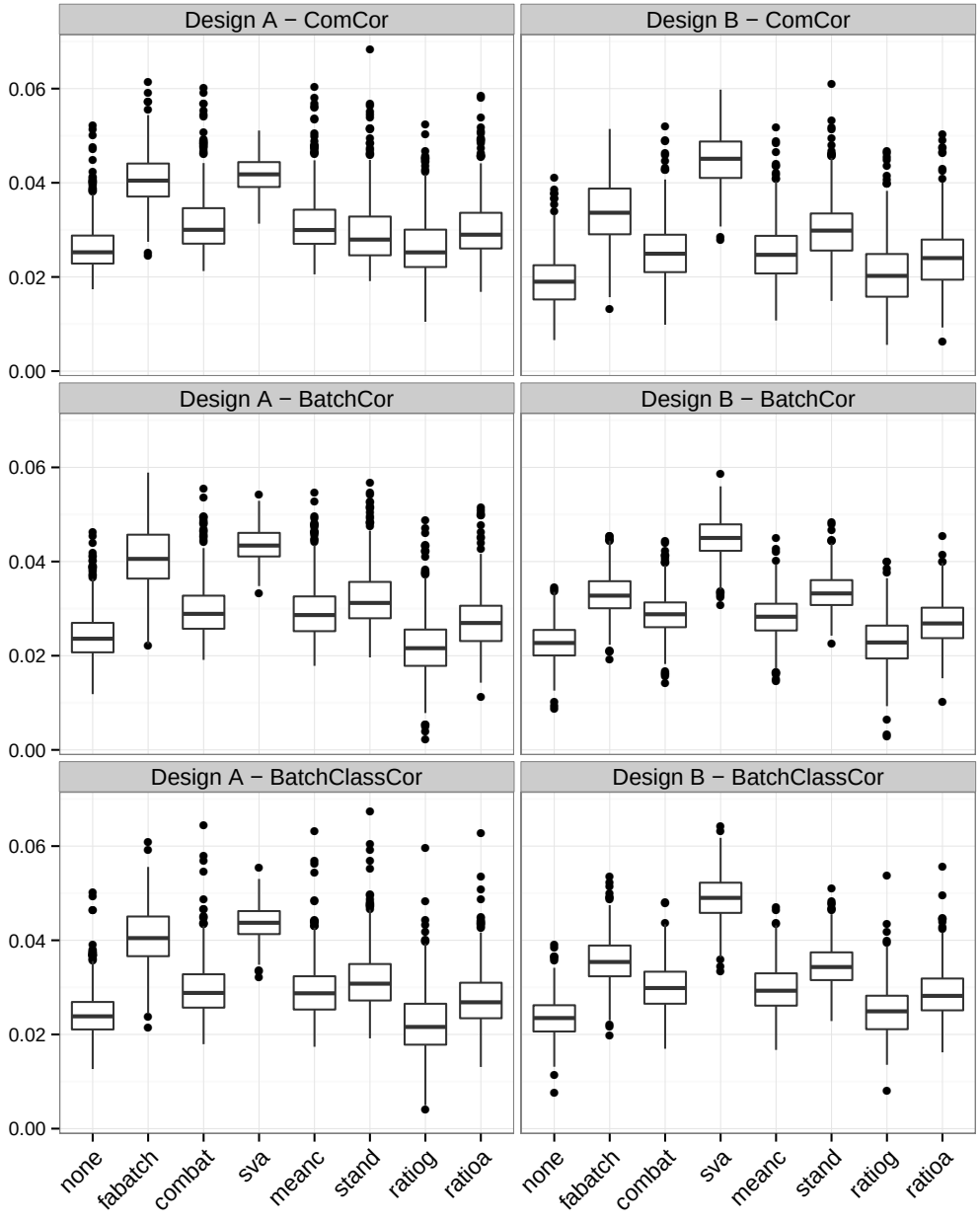

Figure B.10.: Values of metric `pvca` for all simulated datasets separated by simulation scenario and by method.

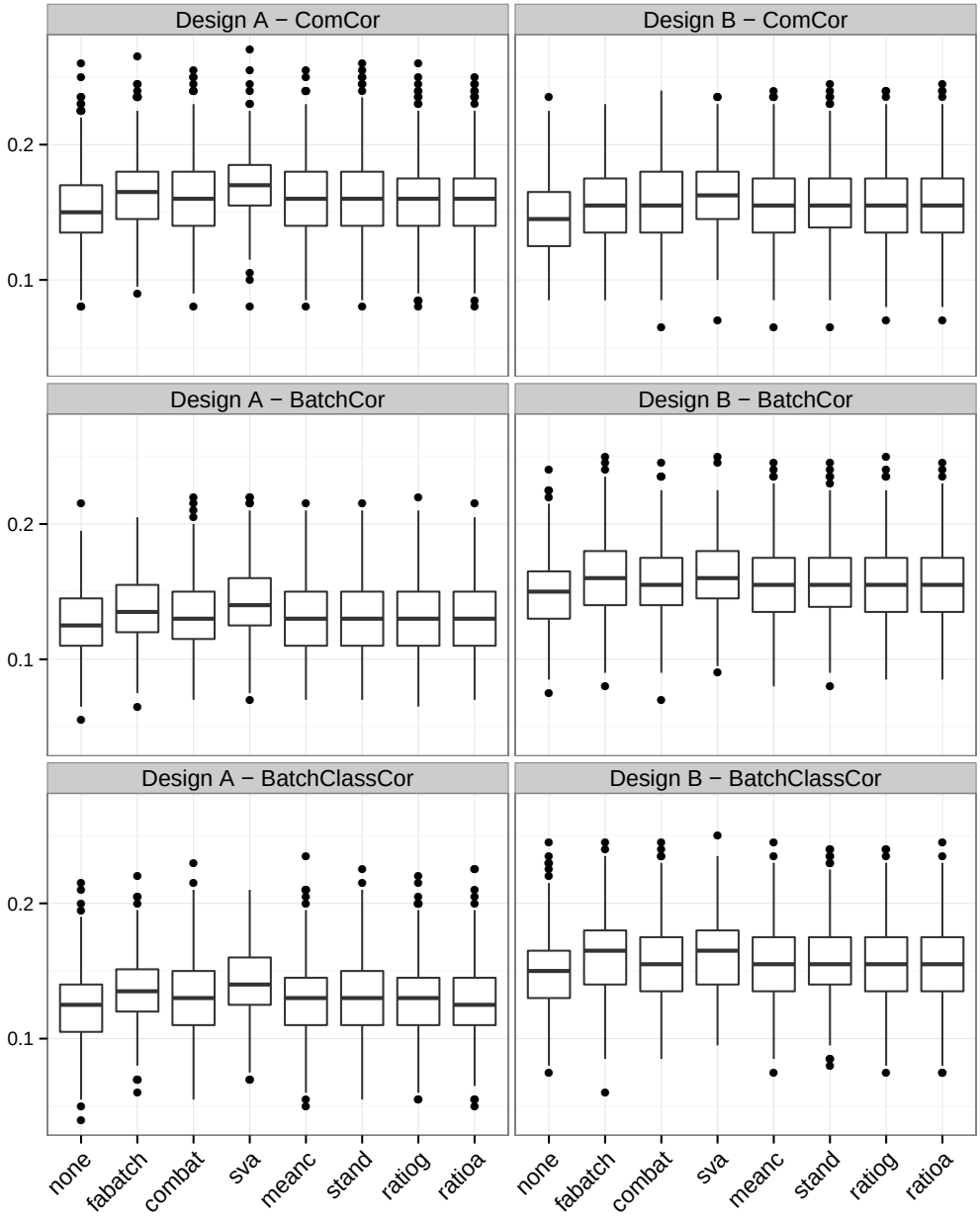

Figure B.11.: Values of metric `diffexpr` for all simulated datasets separated by simulation scenario and by method.

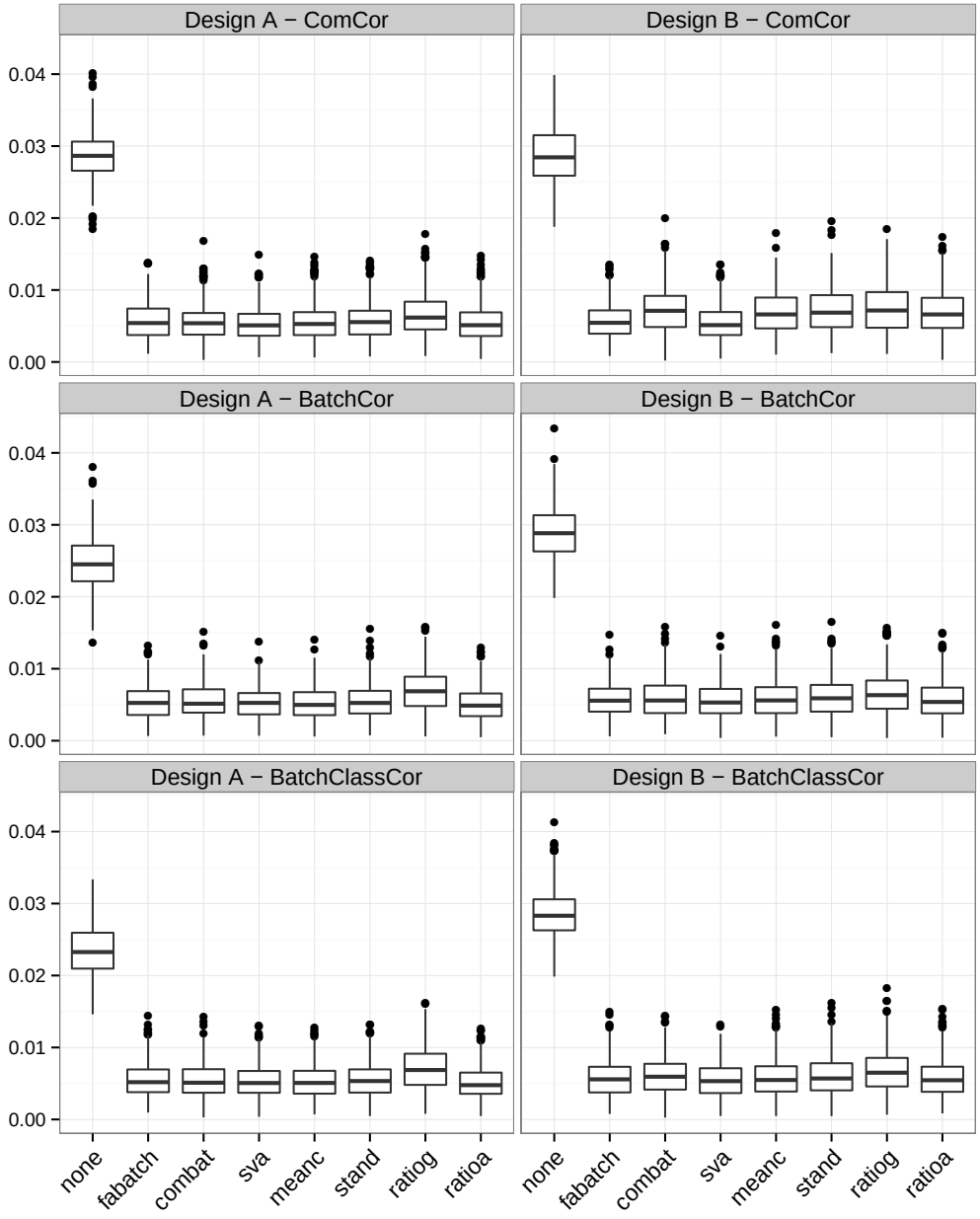

Figure B.12.: Values of metric `skewdiv` for all simulated datasets separated by simulation scenario and by method.

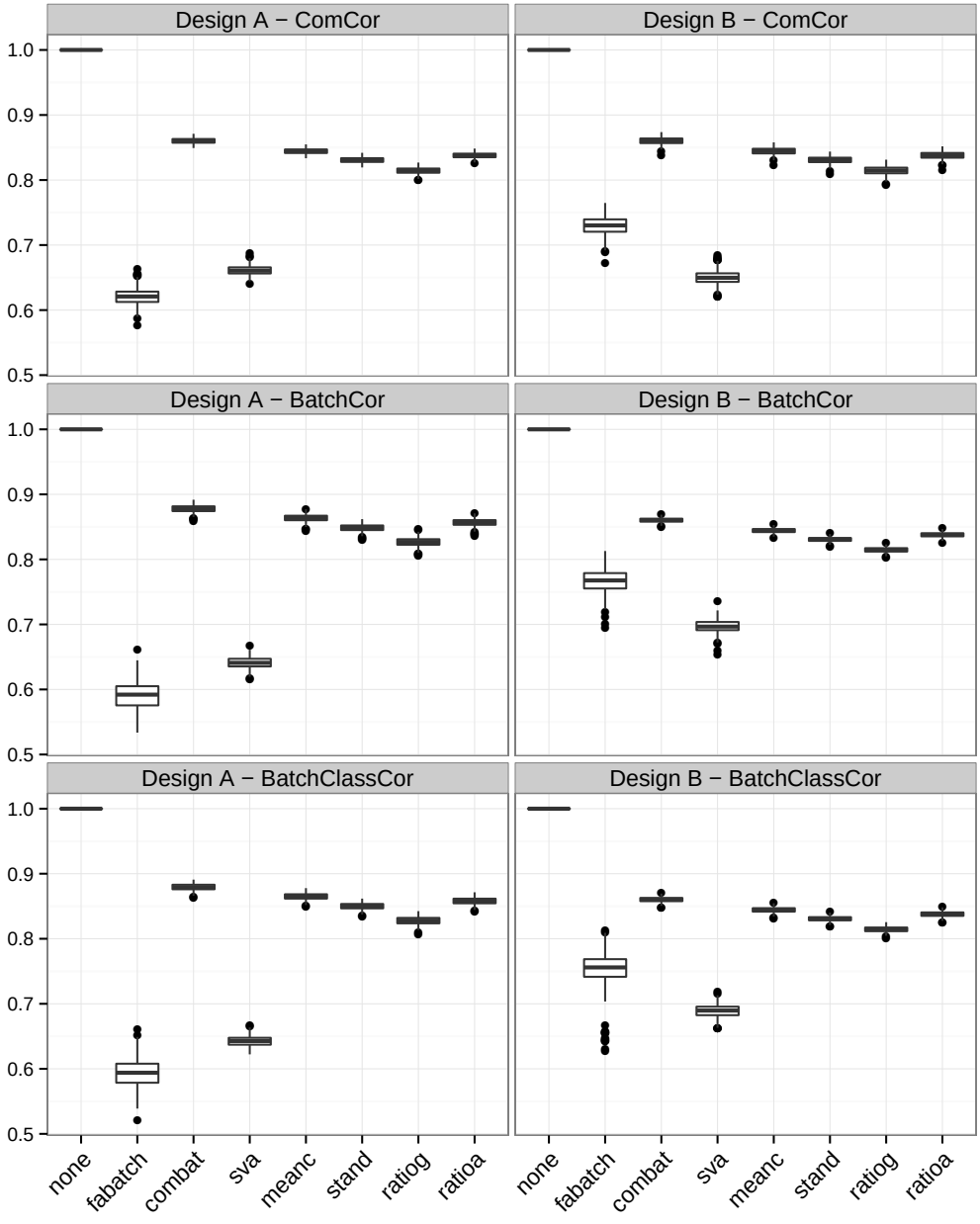

Figure B.13.: Values of metric `corbeaf` for all simulated datasets separated by simulation scenario and by method.

B.5. Tables showing the means of the metric values and of the corresponding ranks in simulated datasets by method and by scenario

Table B.1.: Means of the values of metric `sepscore` and of their ranks among the different methods over all simulated datasets separated by simulation scenario and by method. In each row the results are listed in descending order by mean performance as it relates to the original values and their respective ranks.

Factor-induced correlations - Common correlations

mean values	**fabatch** **0.04259**	combat 0.10329	stand 0.10983	meanc 0.26425	ratiog 0.34545	ratioa 0.36033	sva 0.40368	none 0.41272
mean ranks	**fabatch** **1**	combat 2.272	stand 2.728	meanc 4.012	ratiog 5.159	ratioa 5.851	sva 7.236	none 7.742

Factor-induced correlations - Batch-specific correlations

mean values	**fabatch** **0.03983**	combat 0.06467	stand 0.07055	meanc 0.2652	ratiog 0.33873	sva 0.34239	ratioa 0.3529	none 0.3767
mean ranks	**fabatch** **1.184**	combat 2.168	stand 2.648	meanc 4.022	ratiog 5.807	sva 5.818	ratioa 6.485	none 7.868

Factor-induced correlations - Batch-class-specific correlations

mean values	**fabatch** **0.04157**	combat 0.06515	stand 0.07127	meanc 0.263	ratiog 0.33364	sva 0.34064	ratioa 0.34989	none 0.37793
mean ranks	**fabatch** **1.21**	combat 2.138	stand 2.652	meanc 4.024	ratiog 5.698	sva 5.841	ratioa 6.515	none 7.922

Correlations estimated on real data - Common correlations

mean values	combat 0.09024	**fabatch** **0.09931**	stand 0.11938	sva 0.17365	meanc 0.21915	ratiog 0.32436	ratioa 0.33918	none 0.41279
mean ranks	combat 1.252	**fabatch** **1.825**	stand 2.939	sva 4.056	meanc 4.928	ratiog 6.215	ratioa 6.785	none 8

Correlations estimated on real data - Batch-specific correlations

mean values	**fabatch** **0.08246**	combat 0.09017	stand 0.10397	meanc 0.16851	sva 0.26416	ratiog 0.32547	ratioa 0.34139	none 0.46142
mean ranks	**fabatch** **1.522**	combat 1.742	stand 2.806	meanc 3.932	sva 5.068	ratiog 6.098	ratioa 6.832	none 8

Correlations estimated on real data - Batch-class-specific correlations

mean values	combat 0.10634	stand 0.11988	**fabatch** **0.13248**	meanc 0.146	ratioa 0.28421	ratiog 0.29216	sva 0.30619	none 0.44789
mean ranks	combat 1.444	stand 2.616	**fabatch** **2.809**	meanc 3.139	ratioa 5.574	ratiog 6.072	sva 6.346	none 8

Table B.2.: Means of the values of metric `avedist` and of their ranks among the different methods over all simulated datasets separated by simulation scenario and by method. In each row the results are listed in descending order by mean performance as it relates to the original values and their respective ranks.

Factor-induced correlations - Common correlations

mean values	stand 37.42382	combat 37.53441	meanc 37.58121	ratioa 37.59354	ratiog 37.69106	**fabatch** **40.49195**	none 40.95461	sva 42.10377
mean ranks	stand 1.016	combat 2.212	meanc 3.354	ratioa 3.446	ratiog 4.972	**fabatch** **6.198**	none 6.802	sva 8

Factor-induced correlations - Batch-specific correlations

mean values	ratioa 37.40834	meanc 37.50749	stand 37.50828	ratiog 37.52306	combat 37.57991	none 39.7664	**fabatch** **39.90564**	sva 41.54491
mean ranks	ratioa 1.228	stand 2.768	meanc 2.934	ratiog 3.448	combat 4.622	none 6.354	**fabatch** **6.646**	sva 8

Factor-induced correlations - Batch-class-specific correlations

mean values	ratioa 37.39577	meanc 37.49533	stand 37.49946	ratiog 37.512	combat 37.57149	none 39.78901	**fabatch** **39.93852**	sva 41.58123
mean ranks	ratioa 1.228	stand 2.752	meanc 2.878	ratiog 3.498	combat 4.644	none 6.358	**fabatch** **6.642**	sva 8

Correlations estimated on real data - Common correlations

mean values	stand 37.98839	ratioa 37.98975	meanc 38.01252	combat 38.05134	ratiog 38.09271	**fabatch** **39.49123**	sva 39.82755	none 40.81266
mean ranks	stand 1.764	ratioa 2.064	meanc 2.598	combat 3.982	ratiog 4.592	**fabatch** **6.044**	sva 6.956	none 8

Correlations estimated on real data - Batch-specific correlations

mean values	ratioa 39.74457	ratiog 39.82068	meanc 39.95839	stand 40.1671	combat 40.1761	**fabatch** **40.38741**	sva 40.44444	none 42.62075
mean ranks	ratioa 1.13	ratiog 2.152	meanc 3.112	stand 4.82	combat 5.024	**fabatch** **5.736**	sva 6.026	none 8

Correlations estimated on real data - Batch-class-specific correlations

mean values	ratioa 38.26292	ratiog 38.3701	meanc 38.66762	combat 38.93779	**fabatch** **38.9477**	stand 38.97117	sva 39.28459	none 41.79299
mean ranks	ratioa 1.174	ratiog 2.218	meanc 3.354	**fabatch** **4.69**	combat 4.946	stand 5.468	sva 6.15	none 8

Table B.3.: Means of the values of metric `klmetr` and of their ranks among the different methods over all simulated datasets separated by simulation scenario and by method. In each row the results are listed in descending order by mean performance as it relates to the original values and their respective ranks.

Factor-induced correlations - Common correlations

mean values	combat 0.06141	stand 0.07561	**fabatch** **0.32144**	meanc 0.4242	ratioa 1.34917	ratiog 1.44874	sva 3.78636	none 3.96088
mean ranks	combat 1.344	stand 1.656	**fabatch** **3.164**	meanc 3.836	ratioa 5.002	ratiog 5.998	sva 7.342	none 7.658

Factor-induced correlations - Batch-specific correlations

mean values	combat 0.02796	stand 0.03953	**fabatch** **0.25008**	meanc 0.45892	ratioa 1.41785	ratiog 1.56052	sva 1.66582	none 2.65573
mean ranks	combat 1.344	stand 1.656	**fabatch** **3.036**	meanc 3.964	ratioa 5.198	ratiog 6.376	sva 6.426	none 8

Factor-induced correlations - Batch-class-specific correlations

mean values	combat 0.03325	stand 0.04196	**fabatch** **0.25189**	meanc 0.44475	ratioa 1.41077	ratiog 1.55757	sva 1.61859	none 2.61917
mean ranks	combat 1.394	stand 1.606	**fabatch** **3.05**	meanc 3.95	ratioa 5.2	sva 6.382	ratiog 6.418	none 8

Correlations estimated on real data - Common correlations

mean values	combat 0.03725	stand 0.06081	**fabatch** **0.09927**	sva 0.27584	meanc 0.33733	ratioa 1.32624	ratiog 1.44356	none 3.86244
mean ranks	combat 1.278	stand 1.938	**fabatch** **2.794**	sva 4.286	meanc 4.704	ratioa 6	ratiog 7	none 8

Correlations estimated on real data - Batch-specific correlations

mean values	combat 0.17189	stand 0.19011	**fabatch** **0.23987**	meanc 0.44967	sva 0.69586	ratioa 1.40077	ratiog 1.53715	none 4.57746
mean ranks	combat 1.382	stand 1.888	**fabatch** **2.748**	meanc 4.066	sva 4.922	ratioa 5.994	ratiog 7	none 8

Correlations estimated on real data - Batch-class-specific correlations

mean values	combat 0.14284	stand 0.16164	**fabatch** **0.23314**	meanc 0.35449	sva 0.66427	ratioa 0.94171	ratiog 1.0381	none 3.75182
mean ranks	combat 1.372	stand 1.874	**fabatch** **2.894**	meanc 3.882	sva 5.218	ratioa 5.856	ratiog 6.904	none 8

Table B.4.: Means of the values of metric `pvca` and of their ranks among the different methods over all simulated datasets separated by simulation scenario and by method. In each row the results are listed in descending order by mean performance as it relates to the original values and their respective ranks.

Factor-induced correlations - Common correlations

mean values	sva 0.04174	**fabatch** **0.04079**	combat 0.03155	meanc 0.03139	ratioa 0.03047	stand 0.02972	none 0.02643	ratiog 0.02634
mean ranks	sva 1.69	**fabatch** **1.88**	combat 3.874	meanc 4.086	ratioa 4.752	stand 5.386	ratiog 7.044	none 7.288

Factor-induced correlations - Batch-specific correlations

mean values	sva 0.04362	**fabatch** **0.04099**	stand 0.03228	combat 0.02981	meanc 0.02956	ratioa 0.02743	none 0.02432	ratiog 0.02209
mean ranks	sva 1.482	**fabatch** **1.868**	stand 3.212	combat 4.372	meanc 4.696	ratioa 5.626	none 7.122	ratiog 7.622

Factor-induced correlations - Batch-class-specific correlations

mean values	sva 0.04376	**fabatch** **0.04053**	stand 0.03172	combat 0.02988	meanc 0.02964	ratioa 0.02771	none 0.02459	ratiog 0.02254
mean ranks	sva 1.37	**fabatch** **1.986**	stand 3.436	combat 4.362	meanc 4.604	ratioa 5.568	none 7.154	ratiog 7.52

Correlations estimated on real data - Common correlations

mean values	sva 0.04484	**fabatch** **0.03385**	stand 0.0301	combat 0.02528	meanc 0.02505	ratioa 0.0242	ratiog 0.02065	none 0.01921
mean ranks	sva 1.032	**fabatch** **2.394**	stand 2.932	combat 4.804	meanc 5.032	ratioa 5.276	ratiog 6.96	none 7.57

Correlations estimated on real data - Batch-specific correlations

mean values	sva 0.04494	stand 0.03349	**fabatch** **0.03294**	combat 0.02878	meanc 0.02822	ratioa 0.02702	ratiog 0.0229	none 0.02276
mean ranks	sva 1.008	stand 2.54	**fabatch** **2.888**	combat 4.474	meanc 4.96	ratioa 5.41	ratiog 7.248	none 7.472

Correlations estimated on real data - Batch-class-specific correlations

mean values	sva 0.04893	**fabatch** **0.03542**	stand 0.03463	combat 0.0301	meanc 0.02965	ratioa 0.02862	ratiog 0.02497	none 0.02351
mean ranks	sva 1.008	stand 2.662	**fabatch** **2.76**	combat 4.602	meanc 5.002	ratioa 5.33	ratiog 7.042	none 7.594

Table B.5.: Means of the values of metric `diffexpr` and of their ranks among the different methods over all simulated datasets separated by simulation scenario and by method. In each row the results are listed in descending order by mean performance as it relates to the original values and their respective ranks.

Factor-induced correlations - Common correlations

mean values	sva 0.16971	**fabatch** **0.16407**	combat 0.16098	stand 0.16097	meanc 0.1603	ratioa 0.15957	ratiog 0.15924	none 0.15215
mean ranks	sva 3.192	**fabatch** **3.926**	combat 4.273	stand 4.361	meanc 4.558	ratioa 4.753	ratiog 4.844	none 6.093

Factor-induced correlations - Batch-specific correlations

mean values	sva 0.14241	**fabatch** **0.13769**	combat 0.13267	stand 0.13246	meanc 0.13195	ratioa 0.13094	ratiog 0.13074	none 0.12615
mean ranks	sva 3.202	**fabatch** **3.821**	combat 4.442	stand 4.448	meanc 4.548	ratioa 4.849	ratiog 4.898	none 5.792

Factor-induced correlations - Batch-class-specific correlations

mean values	sva 0.1414	**fabatch** **0.13553**	stand 0.12975	combat 0.12949	meanc 0.12902	ratiog 0.12798	ratioa 0.12777	none 0.12367
mean ranks	sva 2.936	**fabatch** **3.801**	stand 4.477	combat 4.508	meanc 4.618	ratiog 4.909	ratioa 5.015	none 5.736

Correlations estimated on real data - Common correlations

mean values	sva 0.16337	stand 0.15697	combat 0.1568	meanc 0.15587	**fabatch** **0.15579**	ratioa 0.15554	ratiog 0.1552	none 0.14817
mean ranks	sva 3.62	stand 4.134	combat 4.163	meanc 4.433	**fabatch** **4.56**	ratioa 4.572	ratiog 4.64	none 5.878

Correlations estimated on real data - Batch-specific correlations

mean values	sva 0.16086	**fabatch** **0.16064**	combat 0.15724	stand 0.15721	meanc 0.15659	ratiog 0.15598	ratioa 0.15595	none 0.14802
mean ranks	**fabatch** **3.772**	sva 3.791	combat 4.236	stand 4.299	meanc 4.448	ratiog 4.628	ratioa 4.635	none 6.191

Correlations estimated on real data - Batch-class-specific correlations

mean values	sva 0.16286	**fabatch** **0.16221**	stand 0.158	combat 0.15792	meanc 0.15711	ratioa 0.15637	ratiog 0.15621	none 0.14918
mean ranks	sva 3.58	**fabatch** **3.689**	stand 4.235	combat 4.314	meanc 4.528	ratioa 4.755	ratiog 4.803	none 6.096

Table B.6.: Means of the values of metric `skewdiv` and of their ranks among the different methods over all simulated datasets separated by simulation scenario and by method. In each row the results are listed in descending order by mean performance as it relates to the original values and their respective ranks.

Factor-induced correlations - Common correlations

mean values	sva 0.00535	ratioa 0.0054	meanc 0.00547	combat 0.0055	**fabatch** **0.00571**	stand 0.00573	ratiog 0.00651	none 0.02865
mean ranks	ratioa 3.514	meanc 3.696	sva 3.832	combat 3.852	**fabatch** **4.124**	stand 4.252	ratiog 4.73	none 8

Factor-induced correlations - Batch-specific correlations

mean values	ratioa 0.00509	meanc 0.00519	sva 0.00524	stand 0.00543	**fabatch** **0.00544**	combat 0.00556	ratiog 0.00701	none 0.0246
mean ranks	ratioa 3.402	meanc 3.612	sva 3.722	**fabatch** **3.97**	stand 4	combat 4.08	ratiog 5.214	none 8

Factor-induced correlations - Batch-class-specific correlations

mean values	ratioa 0.00515	meanc 0.00527	sva 0.00528	combat 0.00544	**fabatch** **0.00548**	stand 0.0055	ratiog 0.00712	none 0.02349
mean ranks	ratioa 3.486	meanc 3.724	sva 3.796	**fabatch** **3.872**	combat 3.89	stand 4.022	ratiog 5.21	none 8

Correlations estimated on real data - Common correlations

mean values	sva 0.00545	**fabatch** **0.00571**	meanc 0.00692	ratioa 0.00695	stand 0.00716	combat 0.00722	ratiog 0.00751	none 0.02868
mean ranks	sva 3.072	**fabatch** **3.266**	meanc 4.062	ratioa 4.094	combat 4.426	stand 4.432	ratiog 4.648	none 8

Correlations estimated on real data - Batch-specific correlations

mean values	sva 0.00552	ratioa 0.00572	**fabatch** **0.00574**	meanc 0.00584	combat 0.00591	stand 0.00608	ratiog 0.00659	none 0.02892
mean ranks	ratioa 3.592	sva 3.762	meanc 3.832	**fabatch** **3.848**	combat 3.956	stand 4.47	ratiog 4.54	none 8

Correlations estimated on real data - Batch-class-specific correlations

mean values	sva 0.00547	**fabatch** **0.00573**	ratioa 0.00573	meanc 0.00581	stand 0.00601	combat 0.00602	ratiog 0.00674	none 0.0285
mean ranks	ratioa 3.566	sva 3.626	meanc 3.876	**fabatch** **3.95**	combat 4.074	stand 4.222	ratiog 4.686	none 8

Table B.7.: Means of the values of metric `corbeaf` and of their ranks among the different methods over all simulated datasets separated by simulation scenario and by method. In each row the results are listed in descending order by mean performance as it relates to the original values and their respective ranks.

Factor-induced correlations - Common correlations

mean values	none 1	combat 0.86018	meanc 0.84427	ratioa 0.83774	stand 0.83067	ratiog 0.81446	sva 0.66124	**fabatch** **0.62064**
mean ranks	none 1	combat 2	meanc 3	ratioa 4	stand 5	ratiog 6	sva 7	**fabatch** **8**

Factor-induced correlations - Batch-specific correlations

mean values	none 1	combat 0.87781	meanc 0.86351	ratioa 0.85656	stand 0.84845	ratiog 0.82682	sva 0.64111	**fabatch** **0.59071**
mean ranks	none 1	combat 2	meanc 3	ratioa 4	stand 5	ratiog 6	sva 7.006	**fabatch** **7.994**

Factor-induced correlations - Batch-class-specific correlations

mean values	none 1	combat 0.87943	meanc 0.8651	ratioa 0.85816	stand 0.85006	ratiog 0.828	sva 0.64263	**fabatch** **0.59331**
mean ranks	none 1	combat 2	meanc 3	ratioa 4	stand 5	ratiog 6	sva 7.006	**fabatch** **7.994**

Correlations estimated on real data - Common correlations

mean values	none 1	combat 0.86037	meanc 0.84442	ratioa 0.8379	stand 0.83086	ratiog 0.81468	**fabatch** **0.72987**	sva 0.64994
mean ranks	none 1	combat 2	meanc 3	ratioa 4	stand 5	ratiog 6	**fabatch** **7**	sva 8

Correlations estimated on real data - Batch-specific correlations

mean values	none 1	combat 0.86035	meanc 0.84433	ratioa 0.83781	stand 0.83078	ratiog 0.81458	**fabatch** **0.76653**	sva 0.69691
mean ranks	none 1	combat 2	meanc 3	ratioa 4	stand 5	ratiog 6	**fabatch** **7.002**	sva 7.998

Correlations estimated on real data - Batch-class-specific correlations

mean values	none 1	combat 0.86023	meanc 0.84423	ratioa 0.83771	stand 0.83071	ratiog 0.81446	**fabatch** **0.75314**	sva 0.68924
mean ranks	none 1	combat 2	meanc 3	ratioa 4	stand 5	ratiog 6.002	**fabatch** **7.024**	sva 7.974

C. Appendices to Chapter 4

C.1. Plots of the MCC_{rule} values

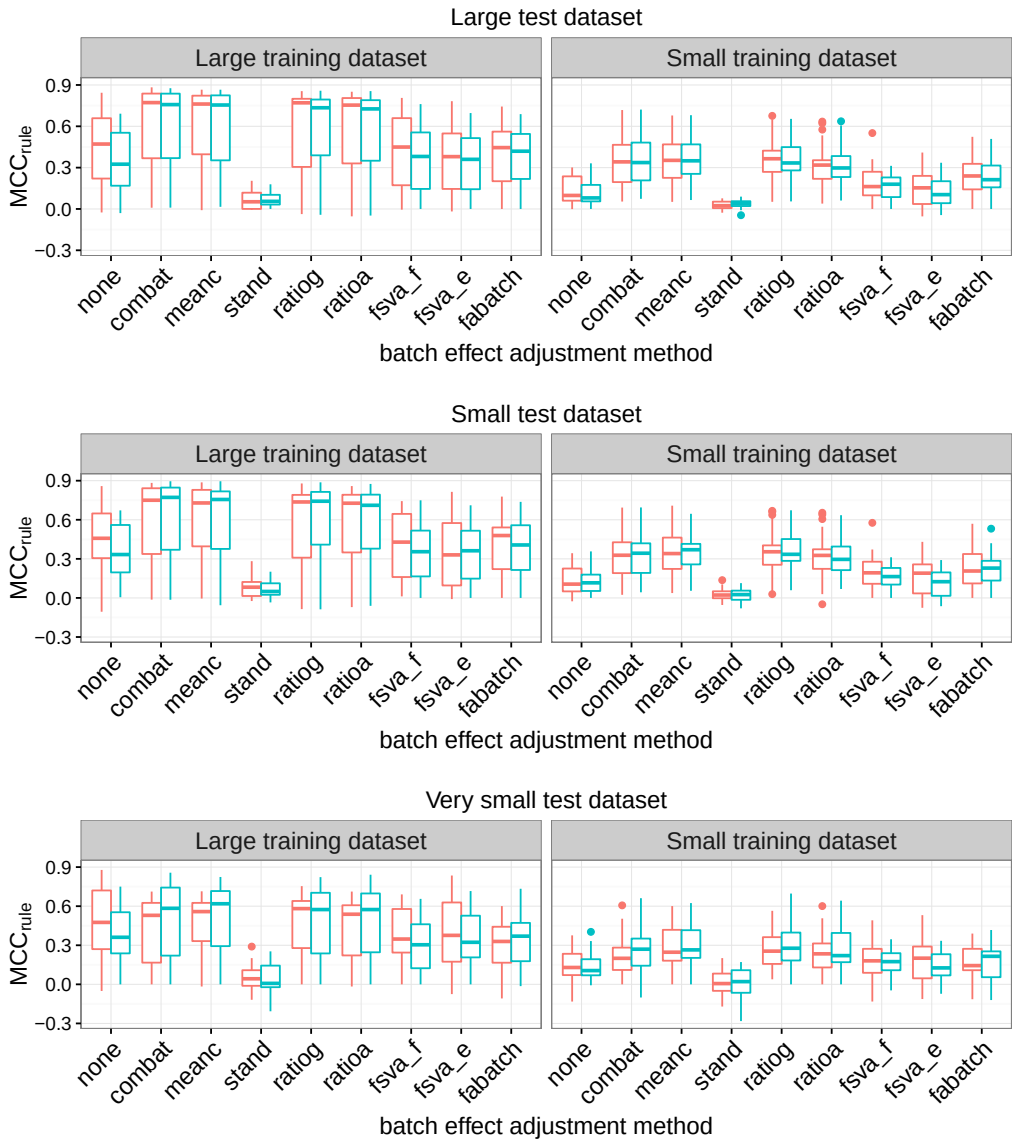

Figure C.1.: MCC_{rule} values for each setting and batch effect adjustment method when using `PLS-LDA` as the classification method. The red and the cyan boxplots show the results when using addon and separate normalization, respectively.

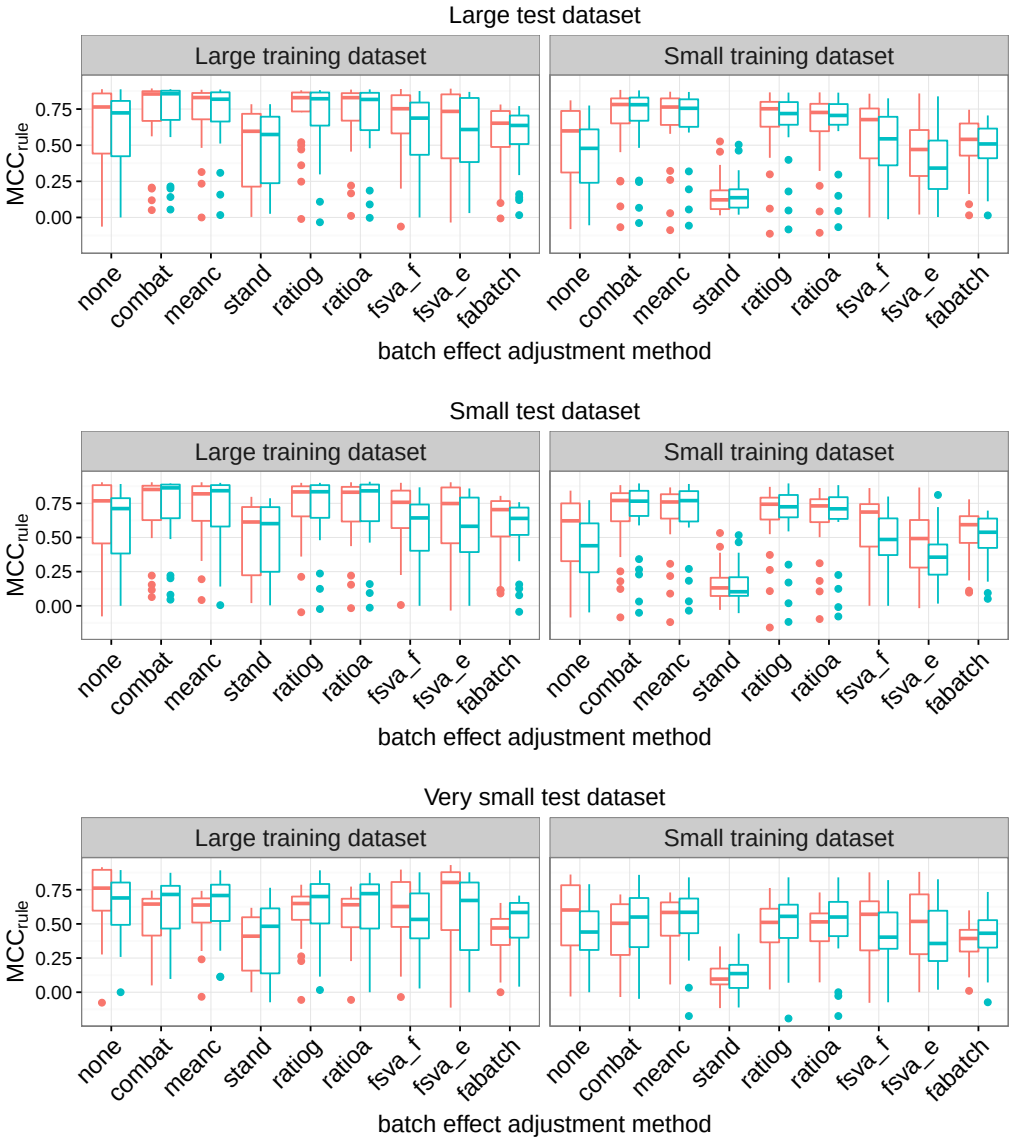

Figure C.2.: MCC_{rule} values for each setting and batch effect adjustment method when using `PLS-LDAvarsel` as the classification method. The red and the cyan boxplots show the results when using addon and separate normalization, respectively.

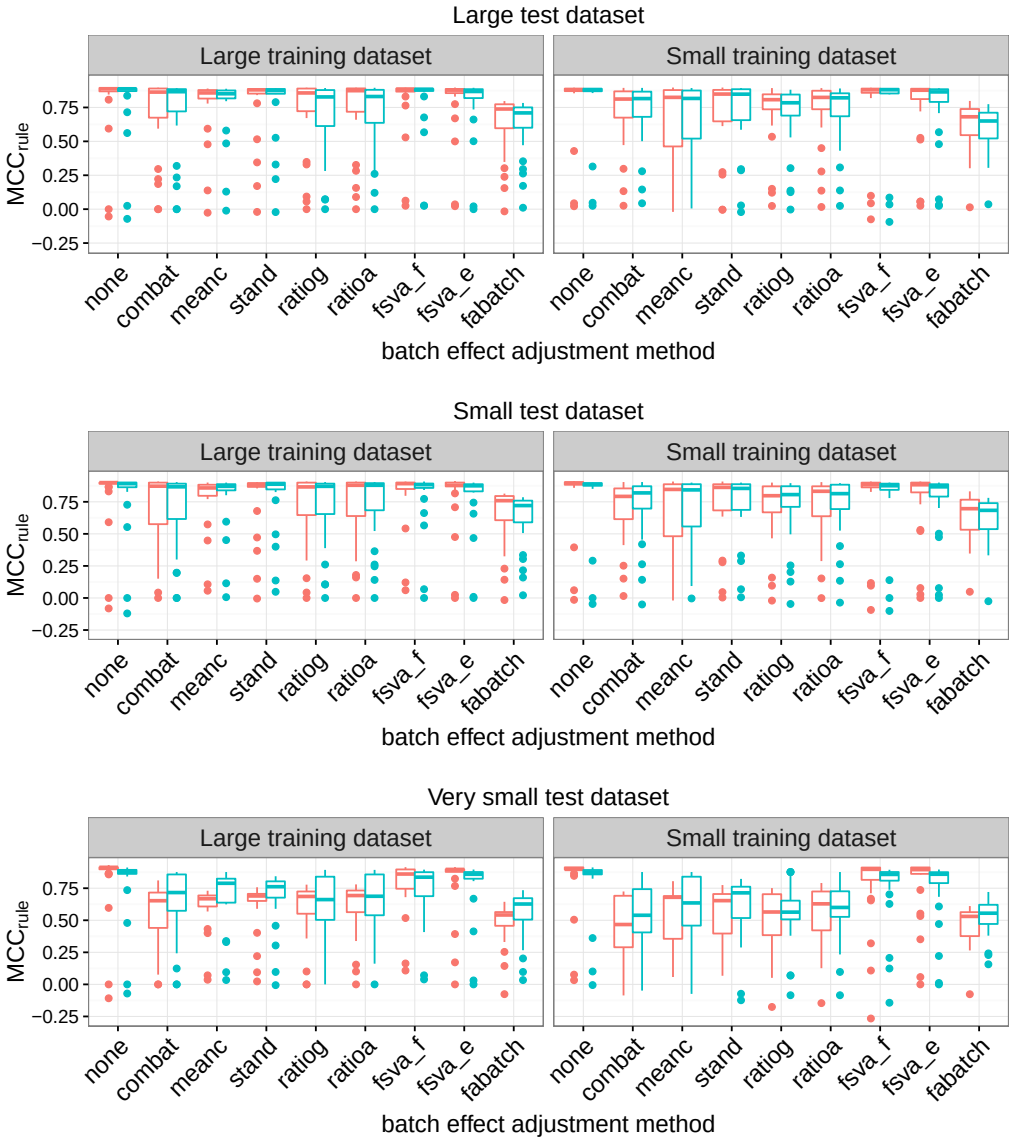

Figure C.3.: MCC_{rule} values for each setting and batch effect adjustment method when using `Boosting` as the classification method. The red and the cyan boxplots show the results when using addon and separate normalization, respectively.

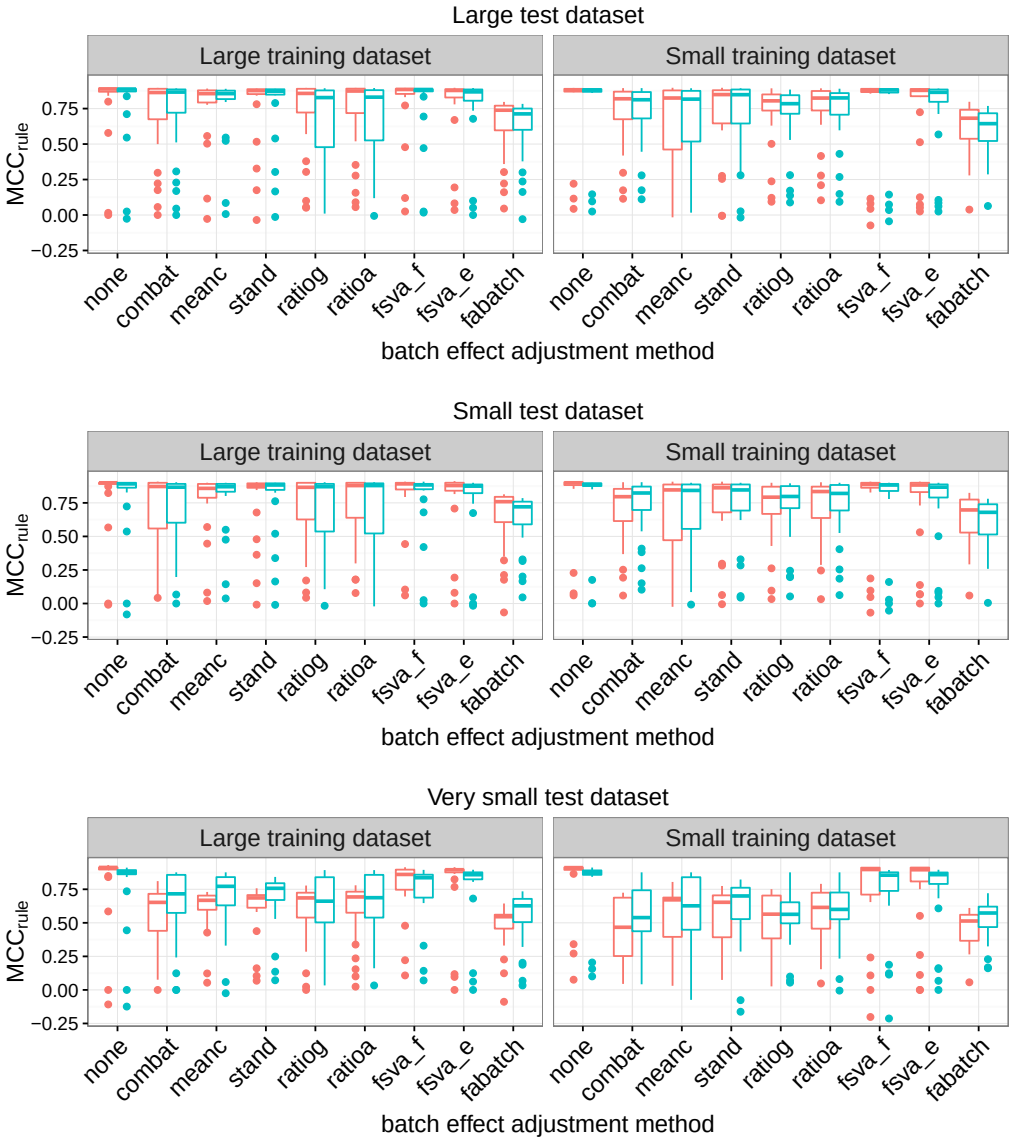

Figure C.4.: MCC_{rule} values for each setting and batch effect adjustment method when using `Boostingvarsel` as the classification method. The red and the cyan boxplots show the results when using addon and separate normalization, respectively.

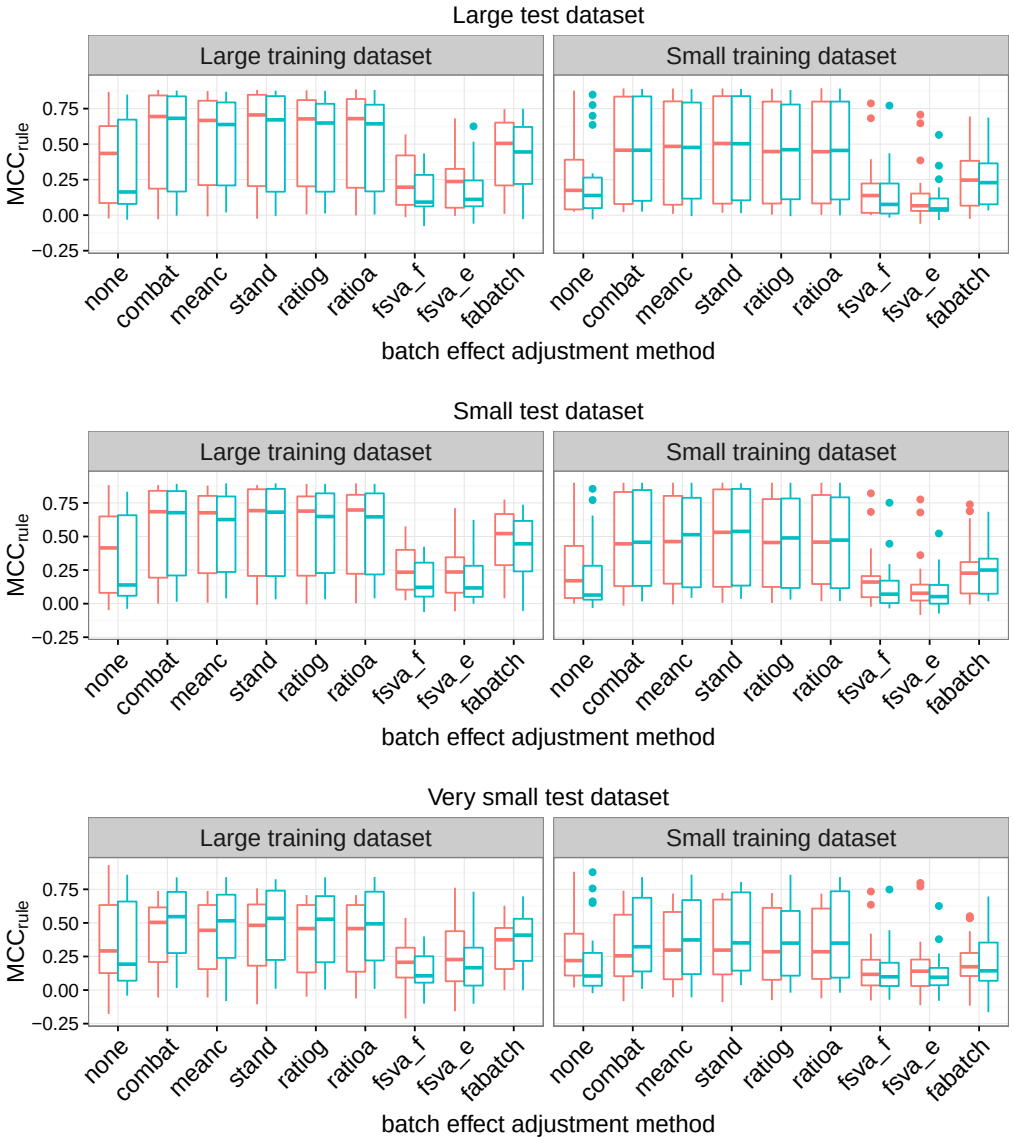

Figure C.5.: MCC_{rule} values for each setting and batch effect adjustment method when using `NSC` as the classification method. The red and the cyan boxplots show the results when using addon and separate normalization, respectively.

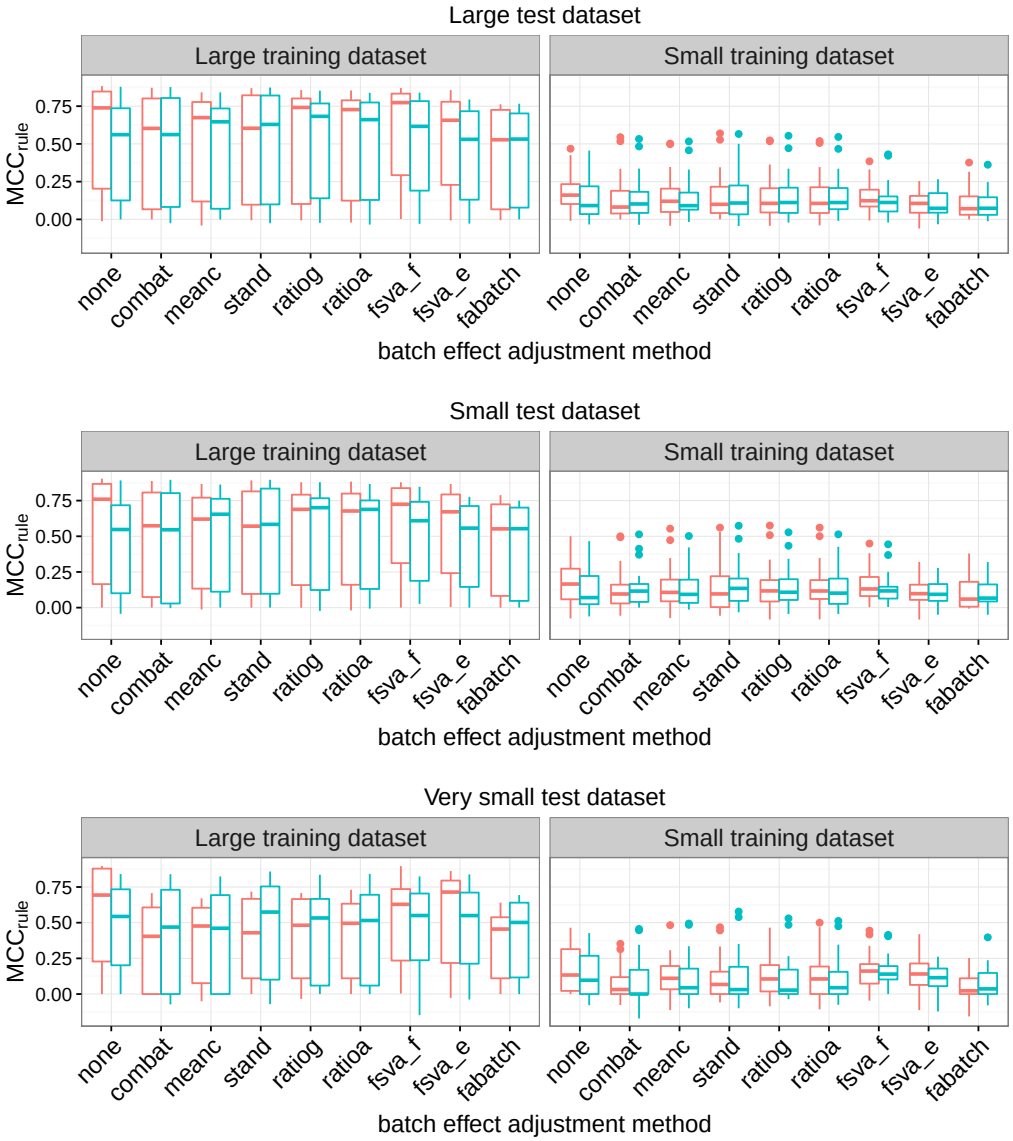

Figure C.6.: MCC_{rule} values for each setting and batch effect adjustment method when using `RF` as the classification method. The red and the cyan boxplots show the results when using addon and separate normalization, respectively.

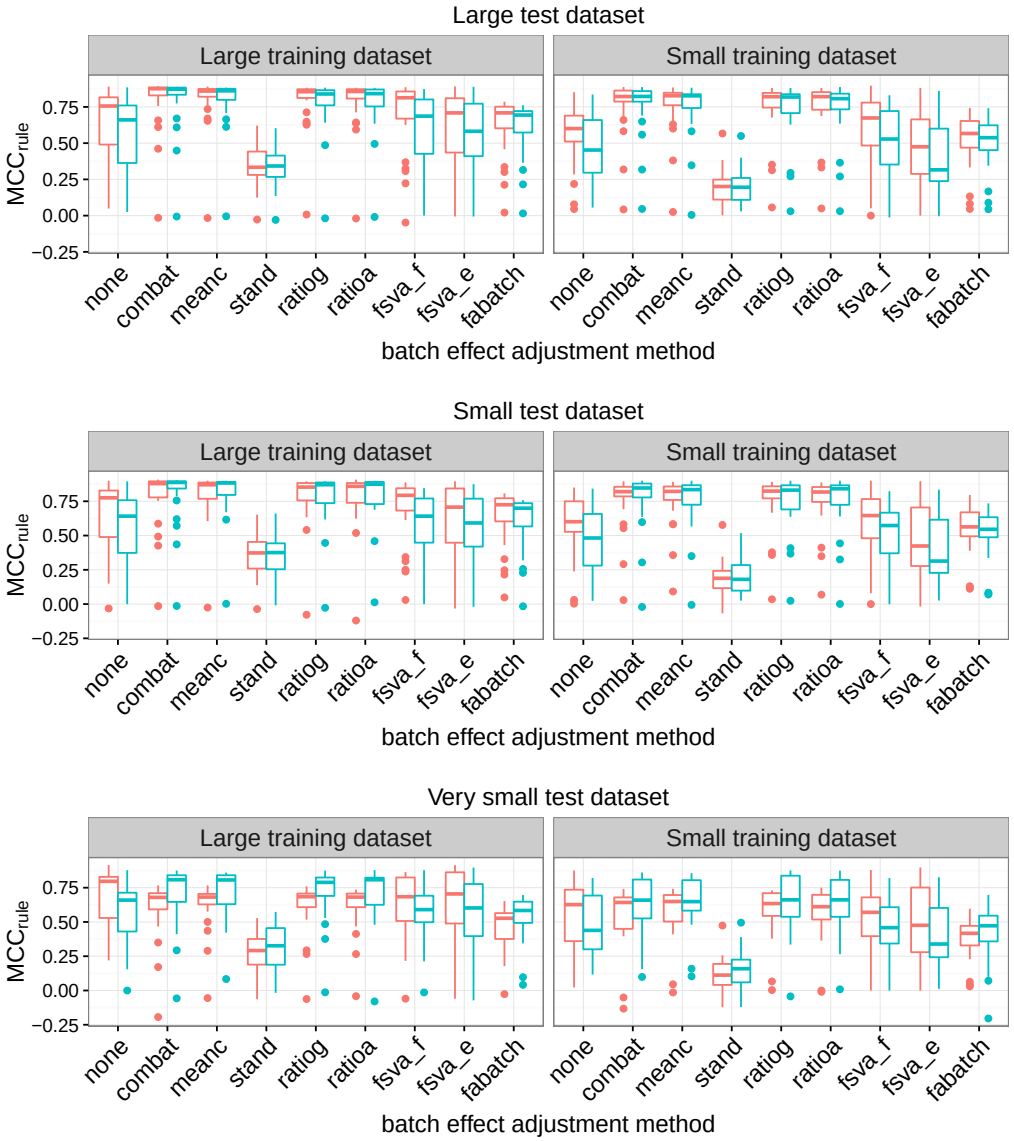

Figure C.7.: MCC_{rule} values for each setting and batch effect adjustment method when using `kNNvarsel` as the classification method. The red and the cyan boxplots show the results when using addon and separate normalization, respectively.

Bibliography

Alemu, E. Y., Carl Jr, J. W., Bravo, H. C. and Hannenhalli, S. (2014). Determinants of expression variability, *Nucleic Acids Research* **42**: 3503–3514.

Ambroise, C. and McLachlan, G. J. (2002). Selection bias in gene extraction on the basis of microarray gene-expression data, *Proceedings of the National Academy of Sciences of the United States of America* **99**: 6562–6566.

Barrett, T., Wilhite, S. E., Ledoux, P., Evangelista, C., Kim, I. F., Tomashevsky, M., Marshall, K. A., Phillippy, K. H., Sherman, P. M., Holko, M., Yefanov, A., Lee, H., Zhang, N., Robertson, C. L., Serova, N. et al. (2013). NCBI GEO: archive for functional genomics data sets–update, *Nucleid Acids Research* **41**: D991–D995.

Bengio, Y. and Grandvalet, Y. (2004). No unbiased estimator of the variance of K-fold cross-validation, *The Journal of Machine Learning Research* **5**: 1089–1105.

Bernau, C., Augustin, T. and Boulesteix, A.-L. (2013). Correcting the optimal resampling-based error rate by estimating the error rate of wrapper algorithms, *Biometrics* **69**: 693–702.

Bernau, C., Riester, M., Boulesteix, A.-L., Parmigiani, G., Huttenhower, C., Waldron, L. and Trippa, L. (2014). Cross-study validation for the assessment of prediction algorithms, *Bioinformatics* **30**: i105–i112.

Bin, R. D., Herold, T. and Boulesteix, A.-L. (2014). Added predictive value of omics data: specific issues related to validation illustrated by two case studies, *BMC Medical Research Methodology* **117**: 14.

Bolstad, B. M., Irizarry, R. A., Åstrand, M. and Speed, T. P. (2003). A comparison of normalization methods for high density oligonucleotide array data based on variance and bias, *Bioinformatics* **19**: 185–193.

Boltz, S., Debreuve, E. and Barlaud, M. (2009). High-dimensional statistical measure for region-of-interest tracking, *Transactions in Image Processing* **18**: 1266–1283.

Boulesteix, A.-L. (2004). PLS dimension reduction for classification with microarray data, *Statistical Applications in Genetics and Molecular Biology* **3**: 33.

Boulesteix, A.-L. (2013). On representative and illustrative comparisons with real data in bioinformatics: response to the letter to the editor by Smith et al., *Bioinformatics* **29**: 2664–2666.

Boulesteix, A.-L. (2015). Ten simple rules for reducing overoptimistic reporting in methodological computational research, *PLoS Computational Biology* **11**: e1004191.

Boulesteix, A.-L., Hable, R., Lauer, S. and Eugster, M. J. A. (2015). A statistical framework for hypothesis testing in real data comparison studies, *The American Statistician* **69**: 201–212.

Boulesteix, A.-L., Janitza, S., Hornung, R., Probst, P., Busen, H., Bischl, B. and Hapfelmeier, A. (in prep). Applicability of prediction rules presented in the literature: a survey on random forest and logistic regression, *Technical report*, Department of Statistics, LMU.

Boulesteix, A.-L., Lauer, S. and Eugster, M. J. (2013). A plea for neutral comparison studies in computational sciences, *PLoS ONE* **8**: e61562.

Boulesteix, A.-L. and Strimmer, K. (2007). Partial least squares: A versatile tool for the analysis of high-dimensional genomic data, *Briefings in Bioinformatics* **8**: 32–44.

Boulesteix, A.-L. and Strobl, C. (2009). Optimal classifier selection and negative bias in error rate estimation: an empirical study on high-dimensional prediction, *BMC Medical Research Methodology* **85**: 9.

Breiman, L. (2001). Random forests, *Machine Learning* **45**: 5–32.

Bühlmann, P. and Hothorn, T. (2007). Boosting algorithms: regularization, prediction and model fitting, *Statistical Science* **22**: 477–505.

Bühlmann, P. and Yu, B. (2003). Boosting with the L2-loss: regression and classification, *Journal of the American Statistical Association* **98**: 324–339.

Bühlmann, P. and Yu, B. (2008). Response to Mease and Wyner, evidence contrary to the statistical view of boosting, *Journal of Machine Learning Research* **9**: 187–194.

Bullard, J. H., Purdom, E., Hansen, K. D. and Dudoit, S. (2010). Evaluation of statistical methods for normalization and differential expression in mRNA-Seq experiments, *BMC Bioinformatics* **11**: 94.

Castaldi, P. J., Dahabreh, I. J. and Ioannidis, J. P. (2011). An empirical assessment of validation practices for molecular classifiers, *Briefings in Bioinformatics* **12**: 189–202.

Chen, C., Grennan, K., Badner, J., Zhang, D., Gershon, E., Jin, L. and Liu, C. (2011). Removing batch effects in analysis of expression microarray data: An evaluation of six batch adjustment methods, *PLoS ONE* **6**: e17238.

Collins, G. S., de Groot, J. A., Dutton, S., Omar, O., Shanyinde, M., Tajar, A., Voysey, M., Wharton, R., Yu, L.-M., Moons, K. G. and Altman, D. G. (2014). External validation of multivariable prediction models: a systematic review of methodological conduct and reporting, *BMC Medical Research Methodology* **40**: 14.

Dai, J. J., Lieu, L. and Rocke, D. (2006). Dimension reduction for classification with gene expression microarray data, *Statistical Applications in Genetics and Molecular Biology* **5**: 6.

Daumer, M., Held, U., Ickstadt, K., Heinz, M., Schach, S. and Ebers, G. (2008). Reducing the probability of false positive research findings by pre-publication validation—experience with a large multiple sclerosis database, *BMC Medical Research Methodology* **18**: 8.

Efron, B. and Tibshirani, R. (1997). Improvements on cross-validation: the .632+ bootstrap method, *Journal of the American Statistical Association* **92**: 548–560.

Friguet, C., Kloareg, M. and Causeur, D. (2009). A factor model approach to multiple testing under dependence, *Journal of the American Statistical Association* **104**: 1406–1415.

Gatto, L., Hansen, K. D., Hoopmann, M. R., Hermjakob, H., Kohlbacher, O. and Beyer, A. (2016). Testing and validation of computational methods for mass spectrometry, *Journal of Proteome Research* **15**: 809–814.

Geman, D., Afsari, B., Tan, A. C. and Naiman, D. Q. (2008). Microarray classification from several two-gene expression comparisons, *in* M. A. Wani, X. W. Chen, D. Casasent, L. A. Kurgan, T. Hu and K. Hafeez (eds), *ICMLA. San Diego*, pp. 583–585.

Geman, D., d'Avignon, C., Naiman, D. Q. and Winslow, R. L. (2004). Classifying gene expression profiles from pairwise mRNA comparisons, *Statistical Applications in Genetics and Molecular Biology* **3**: Article 19.

Geyer, C. J. and Meeden, G. D. (2005). Fuzzy and randomized confidence intervals and p-values (with discussion), *Statistical Science* **20**: 358–387.

Godfrey, A. C., Xu, Z., Weinberg, C. R., Getts, R. C., Wade, P. A., DeRoo, L. A., Sandler, D. P. and Taylor, J. A. (2013). Serum microRNA expression as an early marker for breast cancer risk in prospectively collected samples from the Sister Study cohort, *Breast Cancer Research* **15**: R42.

Hansen, K. D. and Irizarry, R. A. (2012). Removing technical variability in RNA-seq data using conditional quantile normalization, *Biostatistics* **13**: 204–216.

Hastie, T., Tibshirani, R. and Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference and Prediction*, Springer, New York.

Hornung, R., Bernau, C., Truntzer, C., Wilson, R., Stadler, T. and Boulesteix, A.-L. (2015). A measure of the impact of CV incompleteness on prediction error estimation with application to PCA and normalization, *BMC Medical Research Methodology* **15**: 95.

Hornung, R. and Causeur, D. (2015). *bapred: Batch Effect Removal (in Phenotype Prediction using Gene Data)*. R package version 0.1.
URL: *http://cran.r-project.org/package=bapred*

Hornung, R. and Causeur, D. (2016). *bapred: Batch effect removal and addon normalization (in phenotype prediction using gene data)*. R package version 1.0.
URL: *http://cran.r-project.org/package=bapred*

Hsu, C.-W., Chang, C.-C. and Lin, C.-J. (2010). A practical guide to support vector classification, *Technical report*, National Taiwan University.

Huber, W. (2014). Introduction to robust calibration and variance stabilisation with VSN. Vignette. `http://www.bioconductor.org/packages/release/bioc/vignettes/vsn/inst/doc/vsn.pdf`. Accessed 13 Feb 2015.

Huber, W., von Heydebreck, A., Sültmann, H., Poustka, A. and Vingron, M. (2002). Variance stabilization applied to microarray data calibration and to the quantification of differential expression, *Bioinformatics* **18**: S96–S104.

Irizarry, R. A., Hobbs, B., Collin, F., Beazer-Barclay, Y. D., Antonellis, K. J., Scherf, U. and Speed, T. P. (2003). Exploration, normalization, and summaries of high density oligonucleotide array probe level data, *Biostatistics* **4**: 249–264.

Janitza, S., Strobl, C. and Boulesteix, A.-L. (2013). An AUC-based permutation variable importance measure for random forests, *BMC Bioinformatics* **14**: 119.

Johnson, W. E., Rabinovic, A. and Li, C. (2007). Adjusting batch effects in microarray expression data using empirical Bayes methods, *Biostatistics* **8**: 118–127.

Kolesnikov, N., Hastings, E., Keays, M., Melnichuk, O., Tang, Y. A., Williams, E., Dylag, M., Kurbatova, N., Brandizi, M., Burdett, T., Megy, K., Pilicheva, E., Rustici, G., Tikhonov, A., Parkinson, H. et al. (2015). ArrayExpress update–simplifying data submissions, *Nucleid Acid Research* **43**: D1113–D1116.

Kostka, D. and Spang, R. (2008). Microarray based diagnosis profits from better documentation of gene expression signatures, *PLoS Computational Biology* **4**: e22.

Lazar, C., Meganck, S., Taminau, J., Steenhoff, D., Coletta, A., Molter, C., Weiss-Solís, D. Y., Duque, R., Bersini, H. and Nowé, A. (2012). Batch effect removal methods for microarray gene expression data integration: a survey, *Briefings in Bioinformatics* **14**: 469–490.

Lee, J. A., Dobbin, K. K. and Ahn, J. (2014). Covariance adjustment for batch effect in gene expression data, *Statistics in Medicine* **33**: 2681–2695.

Leek, J. T. and Storey, J. D. (2007). Capturing heterogeneity in gene expression studies by surrogate variable analysis, *PLoS Genetics* **3**: 1724–1735.

Li, G.-Z., Zeng, X.-Q., Yang, J. Y. and Yang, M. Q. (2007). Partial least squares based dimension reduction with gene selection for tumor classification, *in* J. Y. Yang, M. Q. Yang, M. M. Zhu, Y. Zhang, H. R. Arabnia, Y. Deng and N. G. Bourbakis (eds), *Proceedings of the 7th IEEE International Conference on Bioinformatics and Bioengineering. Boston*, pp. 1439–1444.

Li, J., Bushel, P., Chu T-M and Wolfinger, R. D. (2009). Principal variance components analysis: Estimating batch effects in microarray gene expression data, *in* A. Scherer (ed.), *Batch Effects and Noise in Microarray Experiments: Sources and Solutions*, John Wiley & Sons, Chichester, UK, pp. 141–154.

Li, J., Liu, Y., Kim, T., Min, R. and Zhang, Z. (2010). Gene expression variability within and between human populations and implications toward disease susceptibility, *PLoS Computational Biology* **6**: e1000910.

Luo, J., Schumacher, M., Scherer, A., Sanoudou, D., Megherbi, D., Davison, T., Shi, T., Tong, W., Shi, L., Hong, H., Zhao, C., Elloumi, F., Shi, W., Thomas, R., Lin, S. et al. (2010). A comparison of batch effect removal methods for enhancement of prediction performance using MAQC-II microarray gene expression data, *The Pharmacogenomics Journal* **10**: 278–291.

Matthews, J. N. S. (2006). *Introduction to Randomized Controlled Clinical Trials*, Chapman & Hall, London, UK.

McCall, M. N., Bolstad, B. M. and Irizarry, R. A. (2010). Frozen robust multiarray analysis (fRMA), *Biostatistics* **11**: 242–253.

Moen, E. L., Zhang, X., Mu, W., Delaney, S. M., Wing, C., McQuade, J., Myers, J., Godley, L. A., Dolan, M. E. and Zhang, W. (2013). Genome-wide variation of cytosine modifications between European and African populations and the implications for complex traits, *Genetics* **194**: 987–996.

Nygaard, V. and Rødland, E. A. (2016). Methods that remove batch effects while retaining group differences may lead to exaggerated confidence in downstream analyses, *Biostatistics* **17**: 29–39.

Nykter, M., Aho, T., Ahdesmäki, M., Ruusuvuori, P., Lehmussola, A. and Yli-Harja, O. (2006). Simulation of microarray data with realistic characteristics, *BMC Bioinformatics* **7**: 349.

Okoniewski, M. J. and Miller, C. J. (2008). Comprehensive analysis of Affymetrix exon arrays using BioConductor, *PLoS Computational Biology* **4**: e6.

Parker, H. S., Bravo, H. C. and Leek, J. T. (2014). Removing batch effects for prediction problems with frozen surrogate variable analysis, *PeerJ* **2**: e561.

Parker, H. S. and Leek, J. T. (2012). The practical effect of batch on genomic prediction, *Statistical Applications in Genetics and Molecular Biology* **11**: Article 10.

Pohjalainen, J., Räsänen, O. and Kadioglu, S. (2015). Feature selection methods and their combinations in high-dimensional classification of speaker likability, intelligibility and personality traits, *Computer Speech & Language* **29**: 145–171.

Rubin, D. B. and Thayer, D. T. (1982). EM algorithms for ML factor analysis, *Psychometrika* **47**: 69–76.

Scheerer, A. (ed.) (2009). *Batch Effects and Noise in Microarray Experiments: Sources and Solutions*, Wiley Series in Probability and Statistics, Wiley, Hoboken.

Schmid, R., Baum, P., Ittrich, C., Fundel-Clemens, K., Huber, W., Brors, B., Eils, R., Weith, A., Mennerich, D. and Quast, K. (2010). Comparison of normalization methods for Illumina BeadChip HumanHT-12 v3, *BMC Genomics* **11**: 349.

Schölkopf, B. and Smola, A. J. (2002). *Learning with Kernels*, MIT Press, Cambridge MA, England.

Seibold, H., Bernau, C., Boulesteix, A.-L. and De Bin, R. (2016). On the choice and influence of the number of boosting steps, *Technical report 188*, Department of Statistics, LMU.

Shabalin, A. A., Tjelmeland, H., Fan, C., Perou, C. M. and Nobel, A. B. (2008). Merging two gene-expression studies via cross-platform normalization, *Bioinformatics* **24**: 1154–1160.

Shenhav, L., Heller, R. and Benjamini, Y. (2015). Quantifying replicability in systematic reviews: the r-value, *arXiv:1502.00088 [stat.AP]* .
URL: *http://arxiv.org/abs/1502.00088*

Simon, R. (2004). When is a genomic classifier ready for prime time?, *Nature Clinical Practice* **1**: 4–5.

Simon, R., Radmacher, M. D., Dobbin, K. and McShane, L. M. (2003). Pitfalls in the use of DNA microarray data for diagnostic and prognostic classification, *Journal of the National Cancer Institute* **95**: 14–18.

Singh, D., Febbo, P. G., Ross, K., Jackson, D. G., Manola, J., Ladd, C., Tamayo, P., Renshaw, A. A., D'Amico, A. V., Richie, J. P., Lander, E. S., Loda, M., Kantoff, P. W., Golub, T. R. and Sellers, W. R. (2002). Gene expression correlates of clinical prostate cancer behavior, *Cancer Cell* **1**: 203–209.

Sonka, M., Hlavac, V. and Boyle, R. (eds) (2014). *Image Processing, Analysis, and Machine Vision*, Cengage Learning, Boston.

Soreq, L., Ben-Shaul, Y., Israel, Z., Bergman, H. and Soreq, H. (2012). Meta-analysis of genetic and environmental Parkinson's disease models reveals a common role of mitochondrial protection pathways, *Neurobiology of Disease* **45**: 1018–1030.

Staaf, J., Vallon-Christersson, J., Lindgren, D., Juliusson, G., Rosenquist, R., Höglund, M., Borg, Å. and Ringnér, M. (2008). Normalization of Illumina Infinium whole-genome SNP data improves copy number estimates and allelic intensity ratios, *BMC Bioinformatics* **9**: 409.

Stein, C. K., Qu, P., Epstein, J., Buros, A., Rosenthal, A., Crowley, J., Morgan, G. and Barlogie, B. (2015). Removing batch effects from purified plasma cell gene expression microarrays with modified ComBat, *BMC Bioinformatics* **16**: 63.

't Hoen, P. A. C., Ariyurek, Y., Thygesen, H. H., Vreugdenhil, E., Vossen, R. H., de Menezes, R. X., Boer, J. M., van Ommen, G. J. and den Dunnen, J. T. (2008). Deep sequencing-based expression analysis shows major advances in robustness, resolution and inter-lab portability over five microarray platforms, *Nucleic Acids Research* **36**: e141.

Tan, A. C., Naiman, D. Q., Xu, L., Winslow, R. L. and Geman, D. (2005). Simple decision rules for classifying human cancers from gene expression profiles, *Bioinformatics* **21**: 3896–3904.

Tibshirani, R., Hastie, T., Narasimhan, B. and Chu, G. (2002). Diagnosis of multiple cancer types by shrunken centroids of gene expression, *Proceedings of the National Academy of Sciences of the United States of America* **99**: 6567–6572.

van't Veer, L. J. and Bernards, R. (2008). Enabling personalized cancer medicine through analysis of gene-expression patterns, *Nature* **452**: 564–570.

Varma, S. and Simon, R. (2006). Bias in error estimation when using cross-validation for model selection, *BMC Bioinformatics* **7**: 91.

Westerhuis, J. A., Hoefsloot, H. C. J., Smit, S., Vis, D. J., Smilde, A. K., van Velzen, E. J. J., van Duijnhoven, J. P. M. and van Dorsten, F. A. (2008). Assessment of PLSDA cross validation, *Metabolomics* **4**: 81–89.

Wong, J. (2013). imputation. R package version 2.0.1.
URL: *http://CRAN.R-project.org/package=imputation*

Wood, I. A., Visscher, P. M. and Mengersen, K. L. (2007). Classification based upon gene expression data: bias and precision of error rates, *Bioinformatics* **23**: 1363–1370.

Xiao, G., Ma, S., Minna, J. and Xie, Y. (2014). Adaptive prediction model in prospective molecular signature-based clinical studies, *Clinical Cancer Research* **20**: 531–539.

Young-Park, M. and Hastie, T. (2007). L1-regularization path algorithm for generalized linear models, *Journal of the Royal Statistical Society B* **69**: 659–577.

Zhu, J. X., McLachlan, G. J., Jones, L. B.-T. and Wood, I. A. (2008). On selection biases with prediction rules formed from gene expression data, *Journal of Statistical Planning and Inference* **138**: 374–386.

Zhu, X., Ambroise, C. and McLachlan, G. J. (2006). Selection bias in working with the top genes in supervised classification of tissue samples, *Statistical Methodology* **3**: 29–41.

Eidesstattliche Versicherung

(Siehe Promotionsordnung vom 12.07.2011, § 8, Abs. 2 Pkt. .5.)

Hiermit erkläre ich an Eidesstatt, dass die Dissertation von mir selbstständig, ohne unerlaubte Beihilfe angefertigt ist.

München, den 19.09.2016 _____

Roman Hornung